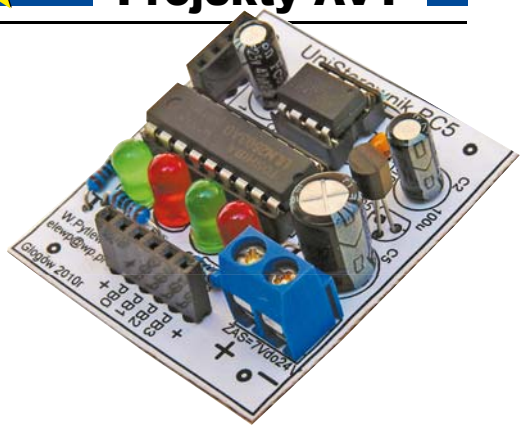
kit 2992 AVT



UniSterownik

część 1 – Dekoder kodu RC-5

Seria projektów **UniSterownik** przeznaczona jest głównie dla tych, którzy do tej pory nie odważyli się na kontakt z mikroprocesorami. W artykule zaproponowany jest interesujący sposób przełamania lodów. Mianowicie nie trzeba pisać kodu – programu dla mikroprocesora. Można ściągnąć z Elportalu gotową, przykładową wersję kodu. Ale lepiej będzie samodzielnie wygenerować „własny" kod na komputerze PC – Autor udostępnił stosowny program. W pierwszej części artykułu szeroko opisany został sposób wpisywania gotowego kodu do procesora.

Dlaczego warto dowiedzieć się, jak zaprogramować mikroprocesor i jak pracują proste układy mikroprocesorowe?

Otóż mikroprocesory są układami niezwykle uniwersalnymi, które poprzez zmianę programu mogą zamieniać dany układ w różne urządzenia. Mikrokontrolery pełnią różnorodne funkcje, zależnie od oprogramowania i możliwości platformy – układu, w którym pracują. Przykładem prostej, uniwersalnej platformy sprzętowej jest prezentowany tu „UniSterownik", sterowany „malutkim" mikrokontrolerem ATTiny13. Dzięki przygotowanym, gotowym programom, każdy Czytelnik może bez trudu przekształcić UniSterownik w układ pełniący funkcje efektów świetlnych, regulatora temperatury, zdalnego sterowania, wskaźnika napięcia czy „inteligentnego" włącznika i to bez użycia lutownicy!

Idea UniSterownika polega na tym, że na początek montujemy płytkę z układem, a potem będziemy zmieniać jego funkcje i przeznaczenie wyłącznie poprzez zmianę programu zawartego w mikrokontrolerze. Istotnym uzupełnieniem będą elementy wykonawcze, podłączane do wyjść płytki: diody LED, silniki, przekaźniki itp. Nasz układ będzie się zmieniał w coraz to inne urządzenie bez potrzeby przerabiania płytki. Co ciekawe, programy dla mikrokontrolera będą generowane przez program *UniSterownik RC5*, uruchomiony na komputerze PC, a użytkownik będzie mógł w prosty sposób dostosować program do swoich upodobań. Wygenerowanym kodem (plik .HEX) będzie trzeba zaprogramować mikrokontroler ATTiny13. Po zamontowaniu zaprogramowanego mikrokontrolera w układzie, UniSterownik będzie pełnił nowe funkcje zgodnie z nowym programem.

Jestem przekonany, że seria artykułów o zastosowaniach UniSterownika zachęci Czytelników do poszerzania swojej wiedzy na temat mikrokontrolerów i sposobów ich stosowania. Mogę zapewnić, że po pokonaniu pierwszych problemów związanych z zaprogramowaniem mikroprocesora, wielu początkujących elektroników „połknie bakcyla" i będzie konstruować układy zawierające mikroprocesory. Na początek trzeba nauczyć się, jak wpisać gotowy program do mikrokontrolera. A to nie jest trudne, jak pokazuje dalsza część artykułu.

Opis układu

Schemat UniSterownika pokazany jest na **rysunku 1**. Układem steruje mikrokontroler ATtiny13. Oczekuje on na sygnał ze scalonego odbiornika podczerwieni US_IR. Nie musimy się zagłębiać w szczegóły. W układzie docelowym mogą być zastosowane różne scalone odbiorniki podczerwieni na 36kHz, np. TFMS5360, SFH5110-36, TSOP1736 lub TSOP31236. Szczególnie polecam układ TSOP31236, ponieważ może on pracować w szerokim zakresie napięć zasilania od 2,5V do 5V. Procesor po odebraniu z wyprowadzenia PB4 prawidłowego sygnału RC5, dekoduje go i zgodnie z otrzymanym rozkazem steruje wyjściami.

Niemniej wejście PB4, połączone ze złączem Z_WE, jest wejściem uniwersalnym i zależnie od konfiguracji programowej przeznaczone jest do odbioru dowolnych sygnałów cyfrowych lub analogowych.

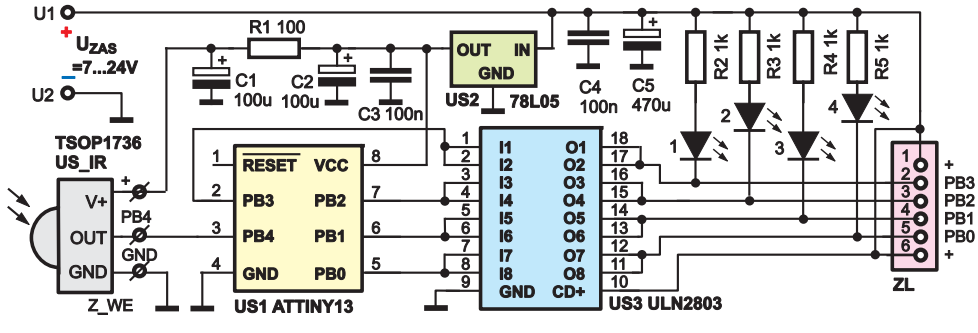
O działaniu układu decyduje program wpisany do mikrokontrolera. Wyjściami sterują wyprowadzenia PB0, PB1, PB2 i PB3. Sygnały z tych wyprowadzeń są wzmacniane poprzez wzmacniacz ULN2803, który zawiera osiem tranzystorów w układzie z otwartym kolektorem o wydajności 0,5A i napięciu do 50V. Przy połączeniu dwóch takich wyjść sterowany prąd może wynosić do 1A. Wyjścia wyprowadzone są na złącze ZL, do którego podłączamy sterowane elementy wykonawcze, czyli małe silniczki, żaróweczki, przekaźniki, itp. Podczas pisania lub analizy programu sterującego należy uwzględnić, że bufory w ULN2803 są negatorami. Ze względu na stabilizator 78L05 napięcie U_{ZAS} nie powinno przekraczać 30V. Diody LED 1, 2, 3, 4 mają zadanie wizualizacji stanu wyjść i są bardzo przydatne podczas pisania i testowania oprogramowania.

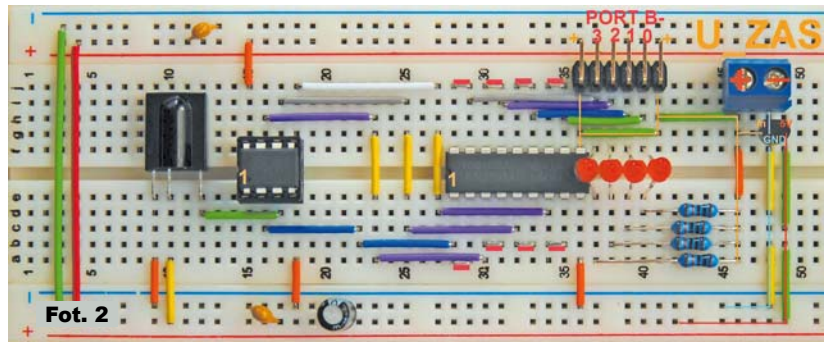
Do sterowania układem UniSterownika ***konieczny jest pilot zdalnego sterowania, pracujący w standardzie RC5***. Kod RC5 został opracowany przez firmę Philips. Duża część pilotów od starszego sprzętu Philipsa powinna prawidłowo współpracować z UniSterownikiem. Inna możliwość to zastosowanie pilota uniwersalnego, który pośród opisów obsługiwanego sprzętu będzie miał wymienione urządzenia firmy Philips i RC5. Standard RC5 jest wykorzystywany także przez inne firmy produkujące sprzęt RTV, dlatego zdobycie takiego pilota nie powinno być problemem.

Montaż i uruchomienie

Układ UniSterownika można zmontować i uruchomić na różne sposoby. Najlepiej

Rys. 1 Schemat ideowy





Fot. 2

zmontować go na płytce drukowanej. Specjalnie opracowaną do tego płytkę widać na **rysunku 2**. Można zakupić kit AVT ze wszystkimi elementami. Ponieważ płytka jest jednostronna, łatwo ją też wykonać samodzielnie, np. metodą termotransferową – pliki projektu umieszczone są w Elportalu wśród materiałów dodatkowych do tego numeru. Jako złącza wejściowe Z_WE i wyjściowe ZL najlepiej użyć dociętych żeńskich gniazd GOLDPIN. Pod procesor ATTiny13 i układ ULN2803 należy wlutować podstawki. Złącze Z_WE ma rozstawione wyprowadzenia tak, że w przypadku montażu odbiornika IR, można wprost w gniazdo włożyć jeden z czterech typów odbiorników podczerwieni (TFMS5360, TSOP1736, TSOP31236 lub SFH5110-36). Odbiornik SFH5110-36 będzie odwrócony, dlatego trzeba go odpowiednio wygiąć. Widok zmontowanej płytki UniSterownika pokazany jest na **fotografii 1**.

Fotografia 2 to widok UniSterownika zmontowanego na płytce stykowej – sposób wykonania, przewidziany dla początkujących i niecierpliwych – układ można uruchomić w przeciągu kilkunastu minut. Do montażu użyłem zworek firmy Velleman z zestawu WJW70. Dla uproszczenia, na płytce stykowej pominąłem elementy R1 i C1, ale w razie złej pracy odbiornika trzeba je dołożyć. Na fotografii dodatkowymi liniami zaznaczyłem krótkie zworki oraz rozprowadzenie początków linii zasilania.

Po złożeniu płytki, ale jeszcze przed zamontowaniem US1, US3, należy sprawdzić wartość napięcia zasilania. Podłączamy do złącza U_ZAS stałe napięcie (biegunowość!) i mierzymy napięcie na wyjściu stabilizatora US2 – powinno wynosić 5V±0,2V. Następnie montujemy bufor ULN2803 i łączymy przewodem „na krótko" wyprowadzenie 8 (+5V) podstawki procesora kolejno z jej wyprowadzeniami 2, 7, 6, 5 – powinny się zaświecić odpowiednio diody LED1, 2, 3, 4. Jeżeli te elektryczne testy wypadły pomyślnie, możemy włożyć do podstawki mikrokontroler. Układ ATTiny13 będziemy wielokrotnie montować i wymontowywać z wlutowanej podstawki, więc po pewnym czasie blaszki jej styków mogą się odgiąć. Dlatego proponuję, żeby do testów używać mikrokontrolera zamontowanego w zwykłej podstawce. Takie rozwiązanie z dwiema podstawkami ułatwia przenoszenie procesora pomiędzy płytką docelową a programatorem i minimalizuje możliwość uszkodzenia mikrokontrolera.

Dekoder kodu RC-5

Pierwszym programem, który uruchomimy na platformie UniSterownika, jest program prostego dekodera kodu RC-5. Umożliwi on przetestowanie świeżo zmontowanego układu. Zadaniem tego programu jest zdekodowanie kodu RC-5 i wskazanie jego „składników", czyli wartości transmitowanego adresu i numeru rozkazu.

Można wykorzystać gotowy program wsadowy RC5.hex, dostępny w Elportalu. Jednak zgodnie z założeniem tej serii projektów, programy dla mikrokontrolera generowane będą raczej przez program *UniSterownik RC5*, uruchomiony na komputerze PC. Program ten został napisany i skompilowany w środowisku Visual Basic. Nie trzeba niczego instalować. Wystarczy skopiować plik *UniSterownik RC5.exe* do komputera i jest on gotowy do pracy. Do poprawnej pracy może być potrzebne uzupełnienie systemu Windows pakietem **FRAMEWORK3,5** - wymagany jest on w starszych wersjach systemu Windows.

Obsługa programu jest prosta. Uruchamiamy w komputerze program *UniSterownik RC5* i otwieramy zakładkę *Dekoder kodu RC5*. Na **rysunku 3** widzimy obraz zakładki programu dekodera. W zakładce możemy wybrać układ pracy dekodera, a następnie wciskamy klawisz opisany tekstem *Generuj program HEX dla ATTiny13*. Program powiadomi nas o wygenerowaniu pliku HEX z danymi do zaprogramowania mikrokontrolera. Otrzymamy również informację o wymaganym ustawieniu fusebitów. Ogólnie aplikacje, które opracowałem do detekcji kodu RC5, powinny mieć zaprogramowane Fusebity na zegar wewnętrzny RC 9,6MHz. Szczegóły w dalszej części artykułu.

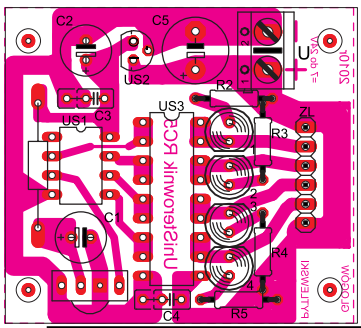
Teraz potrzebny będzie programator, podłączony do komputera PC. Wkładamy mikrokontroler do programatora i programujemy. Obszerne informacje o programatorze i jego obsłudze znajdziesz w dalszej części artykułu.

Po zaprogramowaniu mikrokontrolera montujemy go do podstawki UniSterownika i podłączamy zasilanie.

W oczekiwaniu na transmisję RC5, na diodach LED1, LED2, LED3 i LED4 wyświetlany jest prosty efekt świetlny, sygnalizujący gotowość układu do pracy. Po odebraniu prawidłowego kodu RC5, wszystkie diody LED1...LED4 krótko migną, po czym dłuższymi błyśnięciami kolejno zostaną zaprezentowane wartości kodu. W sumie jeden kod RC-5 to dwie liczby dwucyfrowe. Najpierw dioda LED 1 błyśnie kilka razy, następnie diody LED2...LED4. Jeżeli któraś dioda nie błyśnie, oznacza to wartość 0. Znaczenie liczby błyśnięć diod jest następujące: Liczba błyśnięć LED1 odpowiada liczbie dziesiątek adresu. Liczba błyśnięć LED2 odpowiada liczbie jednostek adresu, liczba błyśnięć LED3 odpowiada liczbie dziesiątek rozkazu, a błyśnięcia LED4 odpowiadają liczbie jednostek rozkazu. Na koniec po „wyświetleniu" danych z kodu RC5 wszystkie diody ponownie krótko migną. Odczytane wartości adresu i rozkazu danego przycisku pilota warto zanotować. Ułatwi to skonfigurowanie sterowania w innych programach, które będą opisywane dla UniSterownika.

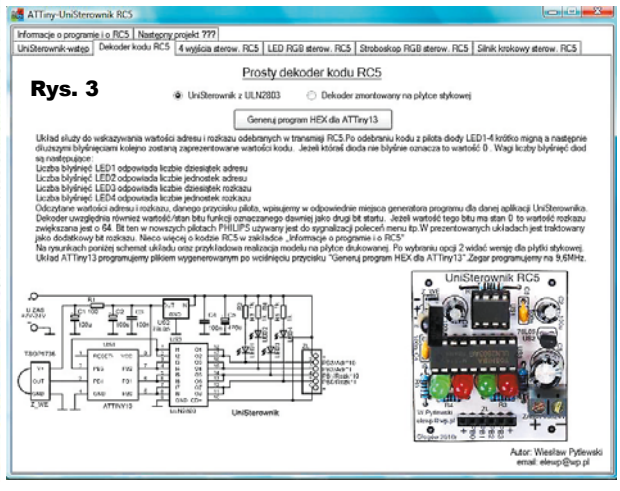
A teraz parę słów wyjaśnienia, o co chodzi z tym rozkazem i adresem zawartym w kodzie RC5. Bity rozkazu zaznaczone na **rysunku 4**, jako „command bits" niosą informację, który przycisk na pilocie został naciśnięty. Natomiast bity adresu opisane jako „system bits" oznaczają, dla jakiego urządzenia przeznaczone są dane. Dzięki temu jednym pilotem możemy sterować wiele urządzeń i jeżeli użyjemy pilota do sterowania tunerem satelitarnym, to nie przełączy nam się program w telewizorze. Na przykład w pilocie uniwersalnym są dostępne przyciski do przełączania-wyboru sterowanych urządzeń TV1,

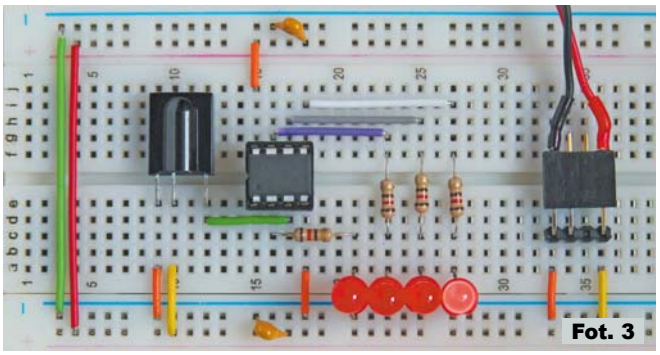
Rys. 2

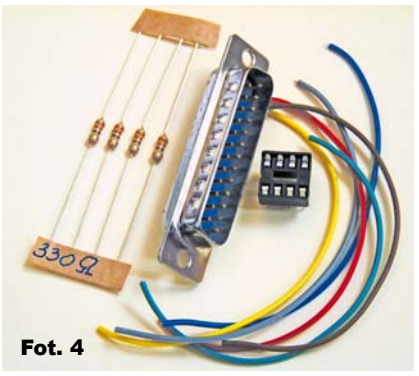


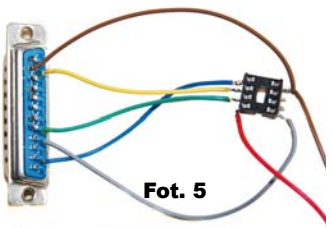
Fot. 1

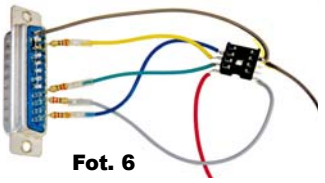
Rys. 3

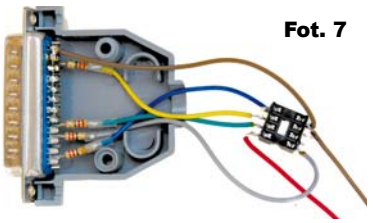


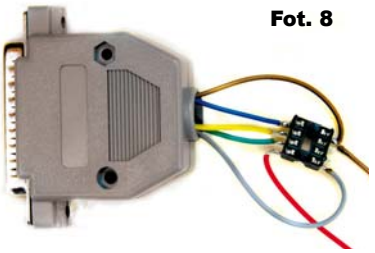

Fot. 3


Fot. 4


Fot. 5


Fot. 6


Fot. 7


Fot. 8

SAT, VCR itd. Poprzez wybór urządzenia zmieniamy adres zawarty w emitowanym kodzie RC5 i w ten sposób możemy wybierać układ UniSterownika, który zareaguje na nasze polecenie. Adresy i rozkazy emitowane przy różnych ustawieniach pilota najlepiej przetestować układem dekodera kodu RC5.

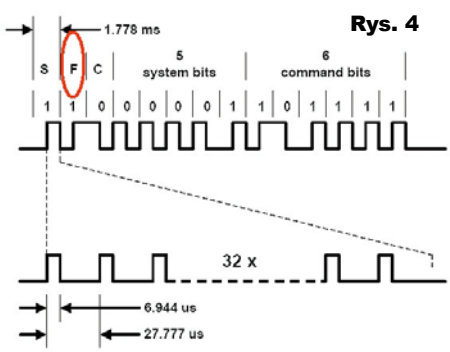
I jeszcze szczegół dla bardziej zaawansowanych: dekoder uwzględnia wartość-stan bitu funkcji, który był oznaczany dawniej jako drugi bit startu. Na rysunku 4 widoczna jest ramka transmisji kodowanej w standardzie RC5 wraz z zakreślonym na czerwono bitem F. Bit ten w nowszych pilotach Philipsa używany jest do sygnalizacji i sterowania poleceń menu. Uznałem, że zgodnie z rozszerzonym zestawem komend RC5 potraktuję ten bit jako dodatkowy – 7 bit rozkazu. Dlatego w prezentowanych zastosowaniach jest uwzględniona jego **zanegowana** wartość, to znaczy, jeżeli wartość tego bitu wynosi 0, to wartość rozkazu zwiększana jest o 64.

Dekoder można też zmontować i uruchomić bez kostki ULN2803. Na **fotografii 3** przedstawiam widok najprostszej wersji dekodera, zmontowanej na płytce stykowej. Płytka jest złożona zgodnie ze schematem z zakładki *Dekoder kodu RC5* w programie *UniSterownik RC5*. Schemat widoczny jest po wybraniu opcji *Dekoder zmontowany na płytce stykowej*.

Programator dla początkujących

Przy pierwszym kontakcie z mikrokontrolerem ogromnym problemem może się wydawać konieczność zaprogramowania układu. Szybko pragnę uspokoić, że ***nie chodzi w tym przypadku o pisanie programu, bo ten uzyskujemy gotowy,*** tylko o wgranie-wpisanie gotowego programu do pamięci procesora. Proponuję, żeby początkujący w dziedzinie programowania mikrokontrolerów postanowili teraz, że nie poddadzą się przy pierwszym niepowodzeniu i konsekwentnie będą próbowali choć raz zaprogramować mikrokontroler. Sam dobrze pamiętam, że początki bywają trudne.

Dla zachęty, przygotowałem opis, jak zbudować i uruchomić prosty programator mikrokontrolerów AVR w wersji Sample Electronics. **Programator ten przy odrobinie wprawy można zmontować i uruchomić w przeciągu kilkudziesięciu minut**. W prezentowanym przykładzie jest on przystosowany do programowania mikrokontrolera ATTiny13, ale łatwo można ten układ przerobić do programowania innych mikrokontrolerów AVR. Wystarczy tylko właściwie przylutować odpowiednią podstawkę. Programator Sample Electronics jest podłączany do równoległego portu drukarkowego w komputerze PC. Port ten określany jest też często jako złącze drukarkowe, port LPT, Parallel Port lub oznaczany jest symbolem drukarki. Dlatego przed rozpoczęciem prac sprawdźmy, czy mamy dostęp do komputera ze złączem LPT i zainstalowanym systemem Windows. Jeżeli nasz komputer nie ma złącza LPT, to trzeba się postarać o dostęp do takiego komputera. Złącza LPT były montowane w starszych modelach komputerów, które dzisiaj mają małą wartość. Może się okazać, że taki komputer stoi niepotrzebny w piwnicy u naszego wujka, znajomego lub sąsiada – warto popytać.



Rys. 4

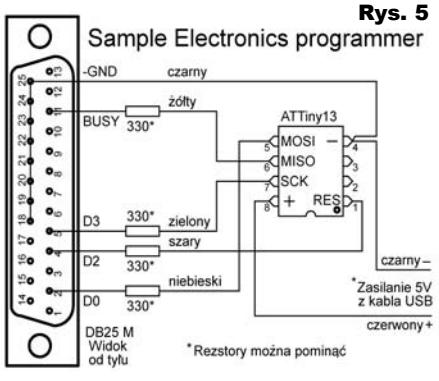
Dane potrzebne do zaprogramowania mikrokontrolera AVR są zawarte w pliku z rozszerzeniem HEX. Plik z danymi HEX musimy przesłać do wnętrza mikrokontrolera. W tym celu wykorzystujemy programator, czyli układ elektryczny, pracujący jako interfejs przekazujący dane z komputera PC do mikrokontrolera. Ostatnim elementem potrzebnym do zaprogramowania mikrokontrolera jest program uruchomiony na PC, który można określić jako program sterujący procesem/przebiegiem programowania. Reasumując, pliki z danymi HEX przesyłamy do mikrokontrolera programatorem. Programator jest sterowany z PC za pomocą programu sterującego przesyłaniem danych.

Montaż programatora

Montaż układu elektrycznego – brzmi to poważnie, ale cała praca polega na polutowaniu kilku kabelków ze złączami, więc następny krok to zebranie części i zmontowanie programatora. Na **fotografii 4** widać elementy potrzebne do budowy programatora. Potrzebujemy złącza DB25M (męskie z wystającymi szpileczkami), kilku przewodów i podstawki DIP8. Złącze DB25M i kabelki można pozyskać z starego kabla drukarkowego. Kabelki powinny być solidne – niezbyt cienkie. Cienkie przewodziki będą się urywać, powodując czasochłonne poszukiwanie przyczyny awarii programatora. Dla zabezpieczenia portu LPT stosujemy rezystory o wartości od 300 do 600Ω. Układ można też zmontować bez rezystorów, ale trzeba go polutować bardzo starannie i uważać podczas użytkowania, żeby nie zrobić zwarcia. Do osłonięcia lutowanych połączeń warto zastosować cienką koszulkę termokurczliwą.

Programator montujemy zgodnie z schematem z **rysunku 5**. Rozpoczynając lutowanie, zwróćmy uwagę na wycięcie z boku podstawki oznaczające pin 1. Wycięcie to jest zaznaczone na dołączonym schemacie układu programatora. Dzięki temu wycięciu wiemy, w jaki sposób ma być ustawiony mikrokontroler podczas montowania go do podstawki. Zasilanie do programowanego mikrokontrolera podłączymy z portu USB. W tym celu musimy zużyć jakiś stary kabel od przedłużacza USB lub np. od myszki, huba, itp. Kabel USB ucinamy tak, żeby złącze A pozostało z długim kawałkiem przewodu. Złącze A to jest to złącze, które wtykamy do portu USB w komputerze. Następnie obieramy koniec przewodu i odcinamy ekran oraz izolujemy przewody mające kolor biały i zielony. Pozostałe dwa przewody czarny (minus) i czerwony (plus) lutujemy do podstawki programatora zgodnie z schematem. Na **fotografiach 5 i 6** przedstawione są dwa modele zmontowanych programatorów.

Przewody użyte do łączenia złącza DB25 z podstawką mikrokontrolera mogą być dłuższe od tych zaprezentowanych na fotografii, ale nie powinny być zbyt długie. Przewody dłuższe niż kilkadziesiąt centymetrów mogą



Rys. 5

być źródłem zakłóceń utrudniających prawidłowe działanie programatora. Na całość zakładamy obudowę złącza **(fotografia 7)** i programator jest gotowy do użytku. Widok na **fotografii 8**.

Uruchomienie i obsługa programu sterującego programowaniem

Do obsługi naszego programatora wykorzystamy oprogramowanie ze środowiska BASCOM-AVR. Jest to przyjazny i intuicyjny program, a jego obsługa jest bardzo prosta. Jeżeli jeszcze nie mamy zainstalowanego programu BASCOM-AVR, to można za darmo pobrać wersję demo tego programu ze strony. ***www.mcselec.com***: *www.mcselec.com/index.php?option=com_docman&task=cat_view&gid=99&Itemid=54*. Prezentowany tu opis przygotowałem na wersji demo 1.11.9.8, ale nowsze wersje nie powinny się niczym różnić w sposobie obsługi programatora Sample Electronics. Instalacja programu przebiega standardowo i wystarczy tylko potwierdzać kolejne pytania instalatora. Warto mieć świadomość, że oprogramowanie BASCOM-AVR to bardzo ciekawe środowisko IDE, umożliwiające pisanie, symulacje i kompilacje programów napisanych w języku BASIC, lecz obecnie interesuje nas tylko opcja programatora i sposób wpisywania danych do układu mikrokontrolera.

Po uruchomieniu programu BASCOM-AVR, wybieramy z paska narzędzi *Options* i następnie *Programmer.* Otworzy nam się okno konfiguracji programatora, widok na **rysunku 6**. W polu wyboru *Programmer* wybieramy *Sample Electronics programmer*. Sprawdzamy również, czy zaznaczone jest pole wyboru opisane jako *AutoVerify*. Opcja ta spowoduje, że po każdym programowaniu będzie sprawdzane czy mikrokontroler ma prawidłowo zapisane dane. Inną istotną opcją jest wybór właściwego adresu portu drukarkowego *LPT-adress,* do którego będzie podłączany programator. Na początek pozostawiamy wartość domyślną 378, ale w razie problemów z pierwszym uruchomieniem programatora może będzie trzeba wybrać inny adres. Po ustawieniu opcji klikamy na OK.

Aby uruchomić okno obsługi programatora, musimy otworzyć dowolny projekt. W tym celu wybieramy z paska narzędzi *File* następnie *Open* i w otwartym okienku wybieramy ścieżkę do pliku np. *BASCOM AVR-SAMPLES-CHIPS-attiny13* lub otwieramy dowolny inny plik z rozszerzeniem BAS. W oknie edytora wyświetli nam się wybrany listing i już możemy uruchomić okno programatora, czyli program, który steruje przepisywaniem danych z dysku komputera PC do mikrokontrolera. Okno programatora uruchomimy, wybierając na pasku narzędzi *Program,* następnie *Send to Chip* i *Manual Program*. Ponieważ nie mamy jeszcze podłączonego programatora, podczas uruchamiania okna wystąpią komunikaty ostrzegawcze. Komunikaty te potwierdzamy, klikając OK.

Proponuję umieścić mikrokontroler w dodatkowej, zwykłej podstawce i dopiero taką „kanapkę" wkładamy w podstawkę programatora – mniej zniszczy się procesor oraz podstawki w programatorze i układzie docelowym. Tuż przed montażem ATtiny13 do programatora i podłączeniem do PC, warto dotknąć metalowej części obudowy komputera PC w celu rozładowania potencjału elektrostatycznego. Wkładając mikrokontroler do podstawki programatora, zwracamy uwagę, aby był on właściwie ustawiony, czyli wycięcie lub kropka oznaczające wyprowadzenie 1 układu pokrywały się z wycięciem oznaczającym 1 na podstawce programatora. Niewłaściwe zamontowanie mikrokontrolera może uszkodzić układ lub programator, dlatego nie należy się spieszyć i wszystko dokładnie sprawdzić. Następnie podłączamy programator do złącza LPT i dopiero na koniec wtykamy kabel zasilania układu do portu USB.

Teraz możemy sprawdzić, czy nasz programator prawidłowo działa. W pasku narzędzi okna programatora wybieramy *Chip,* następnie *Identify*. Jeżeli nie pojawią się żadne komunikaty o błędach i w oknie wyboru mikrokontrolera automatycznie zostanie wybrany ATtiny13, to znaczy, że wszystko jest w porządku i możemy rozpocząć programowanie układu. W razie wystąpienia komunikatów o błędzie trzeba najpierw usunąć problem, korzystając z porad umieszczonych na końcu tego artykułu.

W pamięci mikrokontrolerów AVR są trzy typy wewnętrznej pamięci. Upraszczając nieco, programowana pamięć ma trzy części. Są to pamięć FLASH, pamięć EEPROM i tak zwane Fusebity.

Do pamięci FLASH wpisujemy właściwy program wykonywany przez mikrokontroler. W celu wpisania danych do pamięci FLASH w oknie programatora wykorzystujemy zakładkę *FlashROM*. Do pamięci FLASH programujemy dane z plików BIN lub HEX.

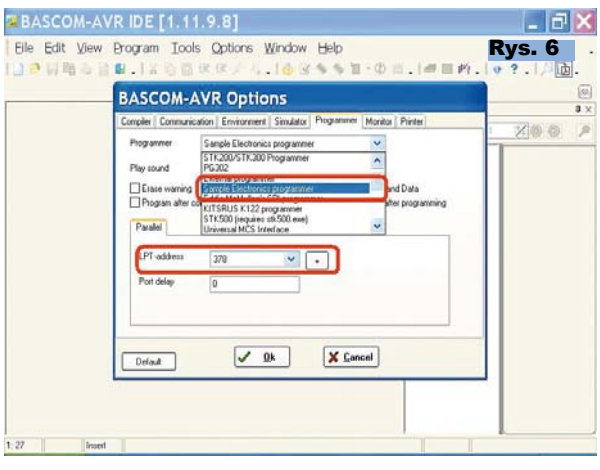
Pamięć EEPROM jest specjalną pamięcią dla ważnych danych. Czasami wymagane jest wpisanie danych do EEPROM podczas programowania mikrokontrolera. Dane takie są umieszczane w plikach z rozszerzeniem EEP. W celu wpisania danych do pamięci EEPROM korzystamy z zakładki *EEPROM*.

W trzecim obszarze pamięci mikrokontrolera, umieszczone są tzw. Fusebity, które przełączają i konfigurują pewne układy wewnątrz mikrokontrolera. Do programowania fusebitów wybieramy zakładkę opisaną jako *Lock and Fuse Bits.* Ważna może być kolejność wpisywania danych do mikrokontrolera. Najpierw programujemy FLASH, a następnie, jeśli jest taka potrzeba programujemy kolejno EEPROM i Fusebity.

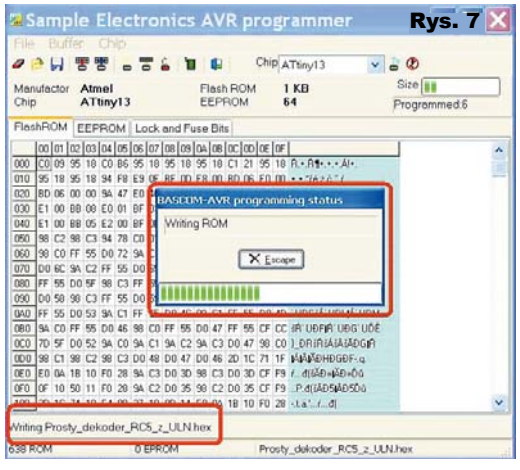
Po prawidłowym zidentyfikowaniu mikrokontrolera przez funkcję *Identify* możemy przystąpić do pierwszej fazy programowania, czyli wpisania programu do pamięci FLASH. Czynność tę można wykonywać wiele razy – około 10000 razy bez negatywnego wpływu na układ i pamięć procesora. Daje to możliwość wypróbowania bardzo wielu ciekawych aplikacji na jednym mikrokontrolerze ATTiny13.

Przed zaprogramowaniem pamięci FLASH, trzeba wskazać programowi, jakie dane chcemy wpisać do mikrokontrolera. W tym celu wybieramy w pasku narzędzi *Buffer*, następnie *Load from file.* Pojawi się okno wyboru pliku. Tu ważna uwaga! W oknie tym musimy zaznaczyć, jaki typ pliku zamierzamy wpisać do bufora programu. Do wyboru mamy ładowanie plików z rozszerzeniem BIN, HEX lub EEP. Pliki generowane przez program UniSterownik mają rozszerzenie HEX, dlatego aby je załadować, koniecznie wybieramy typ HEX. Inaczej pliki nie pokażą się w oknie wyboru. Po wskazaniu pliku jest on ładowany do bufora *FlashROM*. Teraz wreszcie możemy zaprogramować mikrokontroler. Wybieramy na pasku okna programatora *Chip* i następnie *Autoprogram*. W oknie programatora pojawi się pasek przebiegu programowania **(rysunek 7),** a na dolnym pasku z lewej strony pojawi się komunikat o wyniku procesu wpisywania danych do pamięci. Na dolnych paskach umieszczona jest też informacja, jaki plik jest wpisany do bufora programu obsługującego programator.

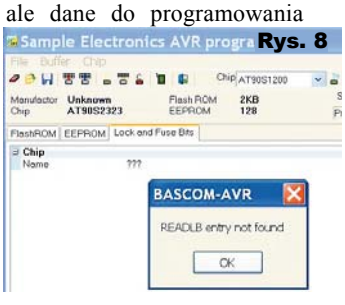
Programowanie pamięci EEPROM przebiega podobnie, ale dane do programowania



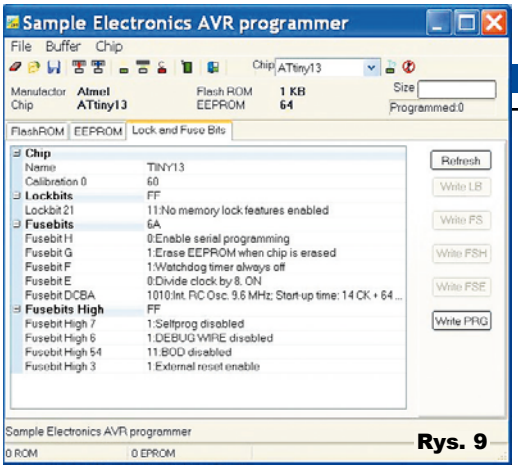
Rys. 6



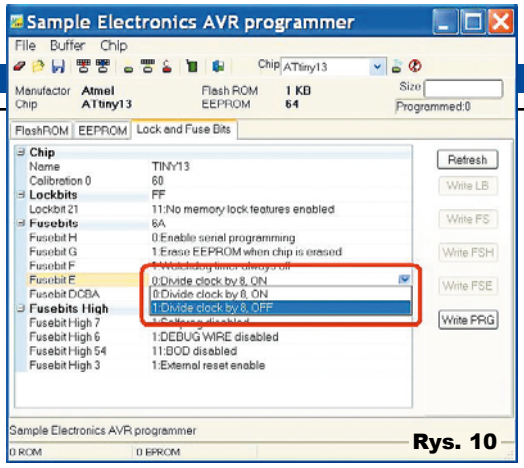
Rys. 7



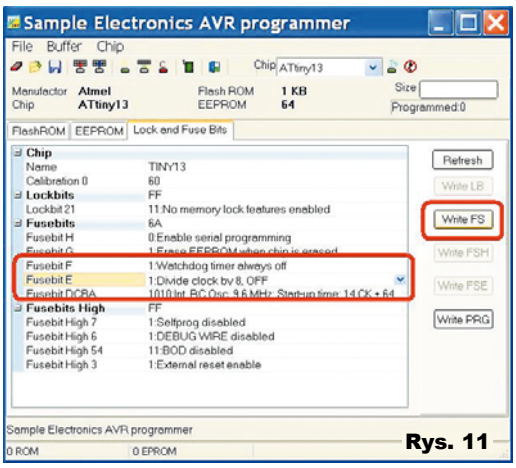
Rys. 8



Rys. 9



Rys. 10



Rys. 11

EEPROM są zazwyczaj zawarte w plikach z rozszerzeniem EEP i oczywiście korzystamy z zakładki EEPROM.

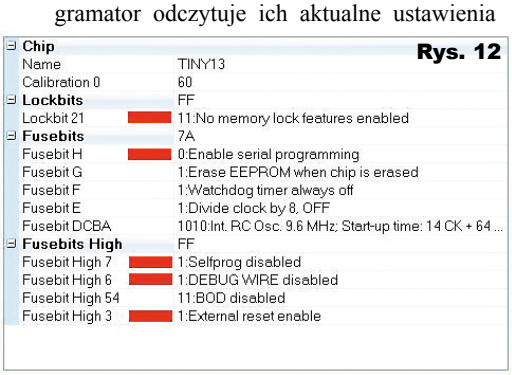
Ostatnia faza programowania to programowanie fusebitów. Jak wcześniej wspomniałem Fusebity są to specjalne „przełączniki" znajdujące się w środku mikrokontrolera, powodujące odpowiednie ustawienie jego wewnętrznych układów. Aby programy działały zgodnie z zamierzeniem programisty, Fusebity muszą być odpowiednio ustawione. Dla właściwej pracy programów generowanych dla platformy UniSterownika, Fusebity powinny ustawiać taktowanie procesora na 9,6MHz oraz wyłączać dzielenie sygnału zegarowego przez 8. Ustawienie innych fusebitów jest w tym przypadku mniej istotne. Nastawy fusebitów nie zmieniają się podczas programowania pamięci FLASH, dlatego nie musimy ciągle ich programować. Dla wszystkich aplikacji/zastosowań UniSterownika wystarczy raz ustawić Fusebity i procesor będzie prawidłowo wykonywał wszystkie proponowane programy.

Aby sprawdzić stan lub przeprogramować Fusebity, w oknie programatora wybieramy zakładkę *Lock and Fuse Bits*. Podczas otwierania zakładki podglądu stanu fusebitów, programator odczytuje ich aktualne ustawienia z mikrokontrolera. Jeżeli odczyt fusebitów zakończy się komunikatem jak na **rysunku 8,** to musimy sprawdzić podłączenie programatora według porad umieszczonych na końcu tego artykułu. Na **rysunku 9** przedstawiony jest prawidłowy odczyt z fabrycznymi ustawieniami fusebitów ATtiny13. Żeby ustawić pracę mikrokontrolera odpowiednio dla UniSterownika, trzeba zmienić Fusebit E, wyłączając dzielenie zegara systemowego przez 8, czyli wybieramy opcję *Divide clock by 8 OFF*. Zmiana opcji w tym programie polega na wybraniu nastawy z rozwijanego menu, w którym opisane są możliwe ustawienia **(rysunek 10)**. Nie musimy od razu wszystkiego rozumieć, ale dla właściwej pracy układu trzeba opcje odpowiednio ustawić. **Bardzo ważne jest, żeby fusebitów nie zmieniać „jak popadnie", ponieważ niektóre z nich przełączają funkcje mikrokontrolera w ten sposób, że po ich przestawieniu niemożliwe jest ponowne zaprogramowanie układu bez użycia specjalnego programatora. Dlatego podczas zmiany ustawień fusebitów trzeba zachować rozwagę i Fusebity ustawić tak, jak zaleca konstruktor układu!** Po każdej zmianie ustawienia danej opcji, po prawej stronie okna programu zostanie podświetlony odpowiedni przycisk, który wprowadza zmiany do pamięci mikrokontrolera **(rysunek 11)**. W ATTiny13 są trzy grupy Fusebitów. Są to *Lockbits*, *Fusebits* i *Fusebits High*. Proponuję po wprowadzeniu jednej lub kilku zmian w danej grupie zmienione dane od razu zapisywać do pamięci. Jeżeli wprowadzimy zmiany w różnych grupach naraz, to nie wszystkie one zostaną zapamiętane. Dlatego zmiany najlepiej wprowadzać grupami. Po wciśnięciu przycisku *Refresh* zostanie odczytany aktualny stan fusebitów z mikrokontrolera, dzięki czemu możemy sprawdzić, czy wszystkie ustawienia są takie, jak zamierzaliśmy. Jeżeli coś niewłaściwie poprzestawiamy w oknie zmian fusebitów i jeszcze dane te nie zostały wprowadzone przyciskiem *Write.*. to możemy się z tego wycofać, wciskając *Refresh* lub wychodzimy z zakładki Fusebitów. Prawidłowe ustawienie fusebitów dla aplikacji UniSterownika przedstawione jest na **rysunku 12**, parametr *Calibration* może się różnić w poszczególnych układach. Pozostałe opcje powinny mieć ustawienia jak na rysunku. Na rysunku tym zaznaczyłem czerwonymi polami bity i opcje, których nie powinniśmy zmieniać bez odpowiedniej wiedzy. Niektóre z tych opcji mogą zablokować pracę mikrokontrolera.

Jeżeli wszystkie opisane wcześniej procedury przebiegną bez błędów, to programowanie mikrokontrolera jest zakończone i można go odłączyć. Odłączenie programatora od komputera wykonujemy w odwrotnej kolejności jak podłączanie, czyli odłączamy kabel zasilania z USB, następnie wyciągamy złącze z portu LPT i wyciągamy mikrokontroler z podstawki programatora.

Jeżeli nie zdecydujesz się, Drogi Czytelniku, na budowę opisanego tu programatora, to pozostaje jeszcze możliwość zakupu. Istnieje mnóstwo typów programatorów. Ja przy opracowywaniu projektów używam obecnie programatora USBASP, ale pierwsze próby z programowaniem AVR-ów robiłem właśnie programatorem Sample Electronics. Po pewnym czasie problemem okazało się to, że w nowszych komputerach nie ma już portu drukarkowego LPT. Wiele informacji na temat programatorów do AVR oraz o sposobie ich wykonania i stosowania można znaleźć w Internecie. Wystarczy wpisać w wyszukiwarce **USBASP, Sample Electronics programmer** czy po prostu **programator AVR**. Dostępne w ofertach programatory można podłączać do komputera PC poprzez złącze USB, LPT, RS lub inne, ale współczesne komputery coraz rzadziej mają wbudowane złącza RS czy LPT. Dlatego najlepiej wybrać programator dołączany do PC poprzez złącze USB.

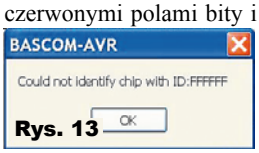
Dostępne na rynku programatory USB, nadające się do zaprogramowania procesorów AVR, nie są drogie i można je nabyć w wielu sklepach internetowych. Powinniśmy wybrać taki programator, na temat którego łatwo znaleźć w Internecie informacje, jak go uruchomić i jak skonfigurować oprogramowanie do jego pracy. Warto zwrócić uwagę na programatory z oferty AVT [*przyp. red.: w następnym numerze EdW zaplanowany jest projekt prostego programatora USB – UniProg*]. Taki jednorazowy wydatek umożliwi nam w przyszłości montowanie i uruchamianie wielu bardzo ciekawych urządzeń wykorzystujących procesory AVR.



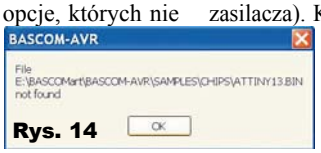
Rys. 12

Problemy z programatorem

Z doświadczenia wiem, że podczas pierwszych prób z nowym typem mikrokontrolera **ZAWSZE** występują jakieś trudności. Przyczyną problemów jest najczęściej niewłaściwa konfiguracja programu ładującego kod do procesora oraz tak banalne sprawy, jak brak zasilania na programowanym układzie, niewłaściwe połączenia czy też złe zamontowanie układu. Odwrotne włożenie procesora ATtiny13 w podstawkę na pewno spowoduje jego bardzo silne rozgrzanie oraz może spowodować uszkodzenie struktury układu scalonego (zależnie od wydajności prądowej zasilacza). Kilka razy zdarzyło mi się popełnić



Rys. 13



Rys. 14

ten błąd, ale na szczęście, układ ATTiny13 po ostygnięciu i właściwym zamontowaniu działał prawidłowo.

A oto ostrzeżenia o błędach, które napotkałem podczas korzystania z Sample Electronics programmer:

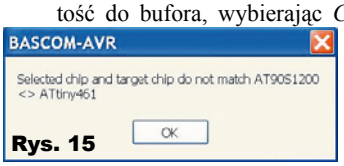
Rysunek 13 – Komunikat informujący o braku możliwości identyfikacji mikrokontrolera. ID o wartości FFFFFF zazwyczaj jest spowodowane brakiem mikrokontrolera w podstawce, a ID o wartości 000000 może oznaczać brak zasilania.

Rysunek 14 – Mało dla nas istotny komunikat, który jest wyświetlany w momencie otwarcia okna obsługi programatora. Informuje o tym, że nie jest dostępny plik wynikowy programu otwartego w oknie edytora środowiska BASCOM-AVR. Jeżeli skompilujemy program otwarty w oknie edytora, to plik wynikowy będzie automatycznie przepisywany do bufora programatora i komunikat nie będzie wyświetlany.

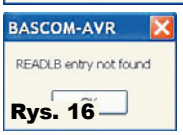
Rysunek 15 – Komunikat informujący o niezgodności typów mikrokontrolerów wybranych w programie i ustawieniach. My i tak musimy wybrać w oknie programatora typ mikrokontrolera, który będziemy chcieli programować, czyli ATTiny13. Ustawienie typu mikrokontrolera można dokonać ręcznie poprzez wybór w oknie programatora opcji z listy *Chip* lub automatycznie poprzez wybór polecenia *Identify*. Jeżeli mikrokontroler będzie w podstawce programatora, to po odczytaniu jego numeru ID zostanie automatycznie wybrany jego typ.

Rysunek 16 – Komunikat, który może wystąpić po wybraniu zakładki *Lock and Fuse Bits*, oznacza błąd podczas próby odczytania stanu fusebitów. Sygnalizuje brak możliwości odczytania danych z mikrokontrolera. Przy takim komunikacie postępujemy według porad poniżej.

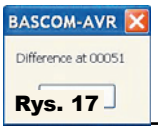
Rysunek 17 – Komunikat ten może wystąpić po programowaniu układu lub podczas porównania danych z mikrokontrolera z danymi z bufora. Komunikat oznacza, że od wskazanego adresu bufora dane w mikrokontrolerze są niezgodne. Jeżeli adres, od którego występuje niezgodność, zmienia się, to znaczy, że w programatorze jest jakiś niepewny styk lub występują zakłócenia. Przy częstym występowaniu tego błędu po wywołaniu *Autoprogram* można spróbować inną metodę. Skasować zawartość mikrokontrolera, wybierając *Chip* i następnie *Erase*. Upewniamy się, czy układ został faktycznie wyzerowany, odczytując jego zawartość do bufora, wybierając *Chip* i następnie *Read chipcode into buffer*. Wszystkie liczby odczytane z wykasowanego mikrokontrolera do bufora FlashROM powinny mieć wartość FF. Następnie ponownie wpisujemy do bufora plik HEX z danymi i kilkakrotnie próbujemy wpisać dane do mikrokontrolera, wybierając polecenie *Write buffer into chip*, po czym weryfikujemy zgodność danych funkcją *Verify*.



Rys. 15



Rys. 16



Rys. 17

W trakcie poszukiwań przyczyn złej pracy programatora wykonujemy test jego pracy przez odczyt Fusebitów – wybieramy zakładkę *Lock and Fuse Bits*. Jeżeli programator odczyta ich ustawienia bez błędów (rysunek 9), to możemy próbować zapisać dane do pamięci FLASH. Zapis tych danych powinien zostać potwierdzony komunikatem *Verified Ok*.

Jeżeli wystąpią problemy z uruchomieniem programatora **przy pierwszym podłączeniu** do komputera, to jeszcze raz sprawdźmy, czy układ jest zlutowany zgodnie ze schematem. Następnie sprawdzamy, czy port LPT właściwie pracuje. W tym celu uruchamiamy specjalne polecenia testujące pracę wyprowadzeń portu. W oknie programatora na pasku narzędzi wybieramy *File*, następnie *Test – Sample Programmer – D0*. Na wyprowadzeniu 5 podstawki mikrokontrolera powinno pojawić się napięcie około 5V. Po wybraniu *D2* napięcie 5V pojawi się na 1 wyprowadzeniu podstawki mikrokontrolera, a na pinie 5 zniknie. Po wybraniu *D3* napięcie 5V pojawi się na 7 wyprowadzeniu. Po wybraniu *Reset*, na wyprowadzeniach 1, 5 i 7 podstawki napięcie powinno wynosić około 0V. Pomiar napięcia na wyprowadzeniu 6 powinien być bliski 5V, ponieważ jest to linia wejściowa portu LPT, podciągnięta rezystorem do 5V. Po wybraniu *Busy* pojawi się komunikat *Busy pin reads 5V*. Następnie zwieramy do minusa linię Busy, czyli pin 6 z pinem 4 podstawki i po wybraniu *Busy* powinien pojawić się komunikat *Busy pin reads 0V*. Pomiary wykonujemy bez mikrokontrolera. Mierzymy również napięcie zasilania na wyprowadzeniach 4(–) i 8(+) podstawki, które po włożeniu wtyczki zasilania do portu USB powinno wynosić około 5V. Jeżeli mamy takie odczyty napięć, to port LPT działa prawidłowo, a przyczyn błędów trzeba szukać w konfiguracji programu obsługującego programator. Sprawdźmy ustawiony typ procesora itp., postępując zgodnie z dalszym opisem.

Jeżeli nie mamy takich napięć i odczytów sprawdźmy, czy programator jest wpięty do właściwego portu LPT – niektóre PC-ty miały dwa lub więcej portów LPT. Spróbujmy również zmienić konfigurację portu na inny adres w opcjach. Zmiany adresu portu LPT dokonujemy w oknie konfiguracji programatora (rysunek 6). Dostępne adresy możemy ustalić, korzystając z menedżera urządzeń.

W Internecie znalazłem też informację, że przyczyną złej pracy programatora może być niewłaściwa konfiguracja portu LPT w BIOS-ie komputera PC. Według tej informacji, port równoległy (Parallel Port) powinien być ustawiony w BIOS-ie na tryb **ECP**, w trybie SPP programator może nie działać.

Jeżeli wystąpią problemy podczas pracy programatora, który działał już na naszym komputerze, to najpierw sprawdzamy:

Czy programator jest wpięty do właściwego portu LPT? Czy jest w podstawce mikrokontroler i czy jest zamontowany we właściwy sposób? Może się też zdarzyć, że wyprowadzenia mikrokontrolera lub podstawki wygięły się i się nie stykają. Sprawdźmy, czy jest podane napięcie zasilania 5V z kabla zasilającego z portu USB? Czy jakieś połączenie lub kabelek nie urwały się? Warto usunąć źródła zakłóceń z okolicy programatora. Mogą to być np. telefon GSM, interfejs Bluetoth lub WIFI itp.

Jeżeli działa funkcja *Identify* i nie możemy zaprogramować układu, to należy sprawdzić, czy dołączone jest zasilanie i szukać przyczyn problemu w obsłudze programu. Sprawdzamy, czy wybrany jest właściwy typ mikrokontrolera w oknie programatora i czy na opisie u góry okna programatora jest nagłówek *Sample Electronics AVR programmer*. Kasujemy pamięć mikrokontrolera, wybierając na pasku *Chip* i *Erase*. Następnie sprawdzamy, czy właściwy plik programu HEX jest wpisany do okna bufora *FlashROM*. Ponownie wywołujemy funkcję *Autoprogram*. Warto od razu zaopatrzyć się w zapasowy mikrokontroler ATTiny13. W razie braku możliwości uruchomienia komunikacji z procesorem zamieńmy układ na zapasowy. W ten sposób możemy wykluczyć uszkodzenie mikrokontrolera jako przyczynę złej pracy programatora. Jeżeli dalej programator nie działa, to wykonajmy opisany wyżej test napięć na porcie LPT.

W przypadku wystąpienia trudności trzeba na spokojnie przeanalizować sytuację i wszystko posprawdzać, ponieważ na 99,9% problem powstał przez nasze niedopatrzenie lub brak wiedzy. W razie kłopotów z uruchomieniem aplikacji na UniSterowniku, proszę pytać. Myślę, że na wiele pytań uda się udzielić odpowiedzi przez Internet.

Praktyka pokazuje, że po pokonaniu pierwszych problemów z konfiguracjami i podłączeniami oraz po zaznajomieniu się z prostymi opcjami oprogramowania, wpisywanie programów do struktury mikroprocesora stanie się łatwą czynnością. **Ta umiejętność otworzy przed nami drzwi do świata wielkich możliwości mikroprocesorów.**

Wiesław Pytlewski
elewp@wp.pl

Wykaz elementów

US1 . . . . . ATTINY13
US2 . . . . . 78L05
US3 . . . . . ULN2803
LED1 do 4 . . dowolne diody LED
US_IR – np. TSOP31236, TFMS5360, TSOP1736 lub SFH5110-36
R1 . . . . . 100Ω
R2,R3,R4,R5 . . . . . 1kΩ
C1,C2 . . . . . 100µF/25V
C3,C4 . . . . . 100nF
C5 . . . . . 470µF/30V
U_ZAS . . . . . Złącze ARK2
Z_WE, ZL . . . . goldpin 2,54mm
Podstawki . . . . . DIP8 i DIP18

Programator

Złącze DB25 Męskie
4 szt. rezystorów 330Ω
3 szt. zwykłych podstawek DIP8

Komplet podzespołów z płytką jest dostępny w sieci handlowej AVT jako kit szkolny AVT-2992.

UniSterownik

część 2 – Zdalnie sterowana dioda RGB

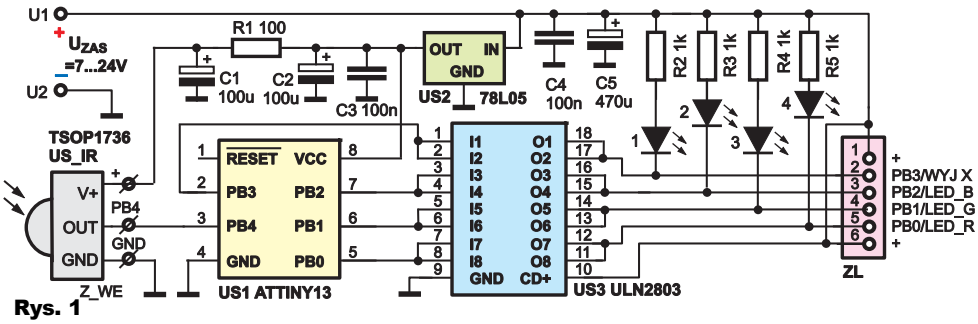
Prezentowana seria projektów **UniSterownik**, rozpoczęta w EdW 11/2011, przeznaczona jest przede wszystkim dla tych, którzy do tej pory nie odważyli się na kontakt z mikroprocesorami. Nie trzeba tu pisać kodu – programu dla mikroprocesora. Można ściągnąć z Elportalu gotową, przykładową wersję kodu. Ale znacznie lepiej będzie samodzielnie wygenerować „własny" kod na komputerze PC – Autor udostępnił stosowny program. Zadaniem początkującego użytkownika jest tylko wpisanie kodu do mikroprocesora i... cieszenie się działaniem tak uzyskanego układu. W części pierwszej artykułu (EdW11/2011), szczegółowo opisano jak zbudować prosty programator i zaprogramować mikroprocesor ATTiny13.

Do czego to służy?

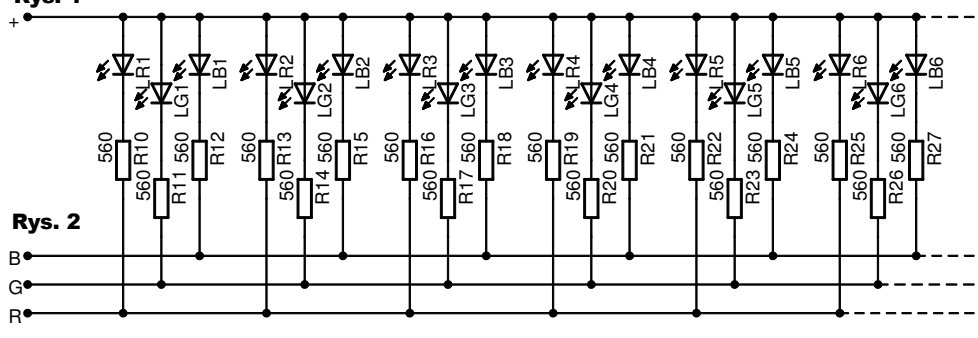
Powszechny dostęp do nowoczesnych diod LED oraz malejące ceny tych elementów spowodowały, że coraz chętniej stosujemy oświetlenie LED. Diody świecące mają wiele zalet, m.in. energooszczędność i długą żywotność. Małe wymiary, różne barwy emitowanego światła i niska temperatura pracy to dodatkowe atuty, powodujące używanie tych diod do dekoracyjnego podświetlania wnętrz i obiektów. Szczególnie interesujące są zespolone, kolorowe diody RGB, które przy odpowiednim zasilaniu wytwarzają światło o dowolnej barwie.

Opisany w EdW 11/2011 UniSterownik pełnić będzie teraz funkcję zdalnego sterowania diody RGB lub taśmy z diod RGB. Układ ma też dodatkowe, zdalnie sterowane wyjście, którym można załączać oświetlenie lub sterować innym urządzeniem.

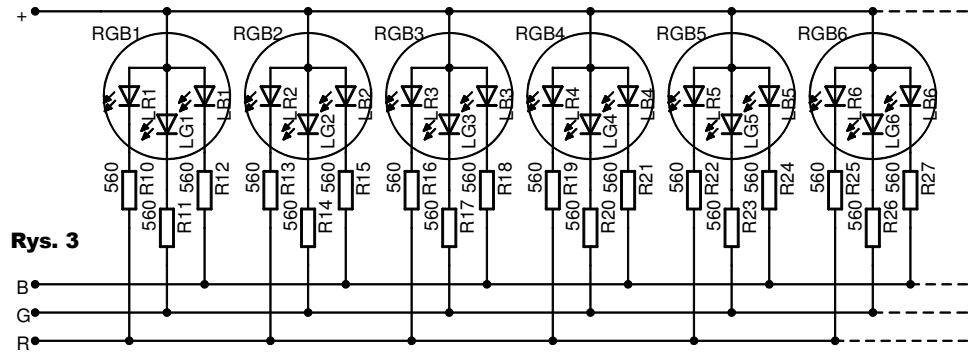
Dzięki udostępnionemu w Elportalu programowi na komputer PC o nazwie „UniSterownikRC5.exe" możemy w prosty sposób wybrać przyciski pilota RC5, które będą uruchamiać poszczególne funkcje sterownika. Program napisany w Visual Basic umożliwia przypisanie adresu i rozkazu RC5 do obsługi poszczególnych funkcji. Takie rozwiązanie ułatwi zbudowanie kilku sterowników, sterujących kilkoma oddzielnymi punktami świetlnymi, które mogą być obsługiwane jednym pilotem.



Rys. 1



Rys. 2

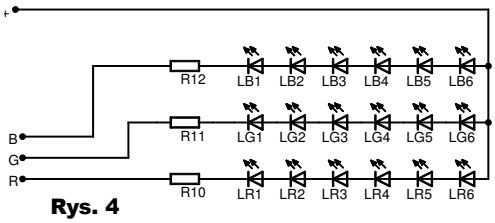


Rys. 3

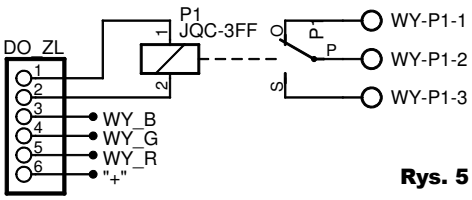
Jak to działa?

Działanie UniSterownika jest szeroko opisane w EdW nr 11/2011. Schemat części sterującej przedstawiony jest na **rysunku 1**. Przypomnę tylko, że kontroler ATTiny13 oczekuje na sygnały z odbiornika podczerwieni np. TSOP1736. Po odebraniu sygnału RC5 procesor zgodnie z otrzymanym rozkazem steruje wyjściami. Do wyjść można dołączać obwody zasilane napięciem stałym do 30V i pobierające prąd do 1A. Diody LED 1, 2, 3, 4 mają zadanie wyświetlać stany napięć na wyjściach i w końcowym układzie mogą być pominięte. Do sterowania jasnością świecenia diod wykorzystana została metoda modulacji szerokości impulsu (PWM).

Aby UniSterownik mógł pełnić funkcję sterownika oświetlenia RGB, trzeba oczywiście odpowiednio dołączyć do niego diody świecące. Mogą to być zarówno diody pojedyncze – świecące w kolorach czerwonym, zielonym i niebieskim, jak i zespolone diody RGB. Do sterowania bezpośrednio z wyjścia UniSterownika nadają się tylko zespolone diody RGB ze wspólną anodą (CA – Common Anode). Poza tym do układu można podłączać modne ostatnio taśmy LED RGB z CA lub oddzielne taśmy, świecące



Rys. 4



Rys. 5

w poszczególnych kolorach. Na **rysunkach 2, 3 i 4** przedstawione są przykładowe schematy podłączenia diod do złącza ZL UniSterownika. Ważne jest, aby nie przekroczyć maksymalnego napięcia pracy i prądu wyjść UniSterownika. Napięcie U_ZAS powinno być takie jak napięcie pracy podłączanych taśm LED. Zazwyczaj jest to 12V. W przypadku potrzeby sterowania większymi obciążeniami, ponad 1A, trzeba zastosować na wyjściu dodatkowe tranzystory mocy.

Wartości rezystorów powinny być odpowiednio wyliczone. Wartość R zależy od napięcia zasilania (U_ZAS) oraz dopuszczalnego prądu diody (I_DIODY) i spadku napięcia (U_DIOD) na świecącej diodzie (rys. 2 i 3) lub na szeregu diod LED (rys. 4). Zależność ta wygląda następująco:

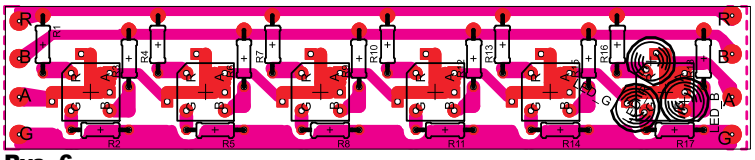
R=(U_ZAS-U_DIOD)/I_DIODY

Szacunkowo można przyjąć, że przy zasilaniu 12V, dla diody o prądzie świecenia równym 20mA, rezystory z rysunków 2 i 3 powinny mieć wartość z przedziału 500Ω do 600Ω. Układ według rysunku 4 wymaga mniej rezystorów, ale diody użyte w takim szeregowym połączeniu powinny być jednakowego typu. Na koniec dwa słowa na temat wyjścia WYJ_X. Wyjście to nie ma sterowania jasnością i może być tylko zdalnie załączane i wyłączane. Ten dodatkowy kanał można wykorzystać do sterowania dodatkowego szeregu diod. Po dołączeniu przekaźnika według schematu z **rysunku 5** wyjście to może sterować konwencjonalnym oświetleniem zasilanym z sieci.

Montaż i uruchomienie

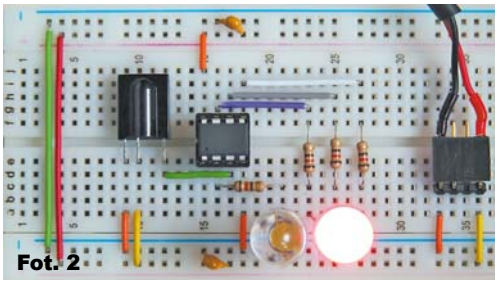
Sposób montażu układu głównego opisałem przy okazji ogólnej prezentacji konstrukcji UniSterownika w EdW nr 11/2011. Do zmontowania oświetlacza z diod LED przydatna może być płytka, której widok przedstawiony jest na **rysunku 6**. Płytka ta nadaje się zarówno do montażu zwykłych diod o średnicy 5mm, jak i zespolonych diod RGB typu SUPER FLUX. Sposób montażu różnych typów diod widać na **fotografii 1**. Na **rysunku 7** widoczna jest płytka przewidziana do montażu przekaźnika (JQC-3FF na 12V lub 24V) sterowanego z dodatkowego wyjścia WYJ_X.

Na **fotografii 2** przedstawiony jest widok najprostszej wersji sterownika pojedynczej diody RGB, wykonanego na płytce stykowej. Płytka jest zmontowana zgodnie ze schematem z zakładki *„LED RGB sterow. RC5”* w programie *„UniSterownik RC5”*. Schemat widoczny jest po wybraniu opcji *„Sterowany LED RGB - WA”*. Na płytce stykowej można również przetestować działanie układu z zespoloną diodą RGB w układzie z wspólną katodą (WK lub CC). Wybieramy wtedy w programie opcje *„Sterowany LED RGB – WK”* i programem wygenerowanym w tej opcji, programujemy procesor. Dioda po lewej stronie załączana jest poleceniami sterującymi wyprowadzeniem WYJ_X.



Rys. 6

Fot. 1



Fot. 2

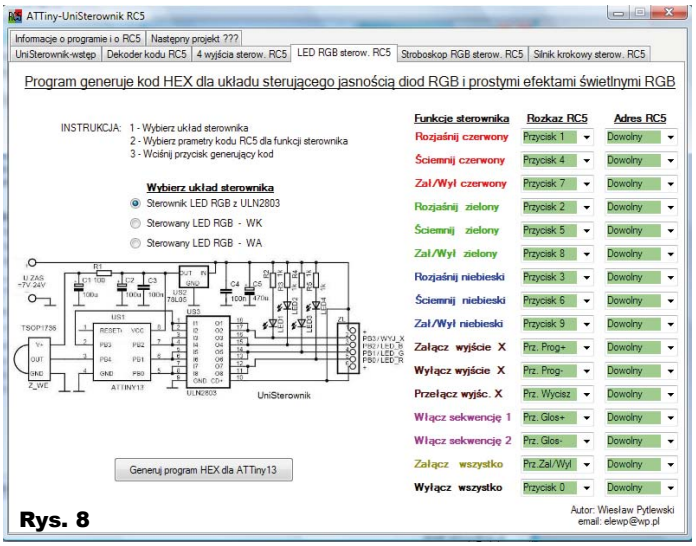
Wsad HEX do zaprogramowania mikrokontrolera ATTiny13 generowany jest przez program *„UniSterownik RC5”*. Program ten powstał w celu umożliwienia zmiany przycisków pilota, sterujących poszczególnymi funkcjami sterownika. Rozkaz i adres generowany po przyciśnięciu danego przycisku w pilocie, można ustalić za pomocą aplikacji prostego dekodera kodu RC5, opisanego w EdW nr 11/2011.

Uruchamiamy program na PC i wybieramy zakładkę *„LED RGB sterow. RC5”*. Obraz okna tej zakładki widoczny jest na **rysunku 8**. Teraz możemy skonfigurować przyciski zgodnie z naszymi potrzebami. Poszczególnym funkcjom sterownika przypisujemy przyciski pilota RC5, definiując ich rozkaz i adres. Funkcje realizowane przez UniSterownik pracujący jako sterownik diod RGB są intuicyjnie opisane w programie konfigurującym UniSterownikRC5. Funkcje *„Rozjaśnij”*, *„Ściemnij”* i *„Zal/Wył”* powodują odpowiednie zmiany w świeceniu poszczególnych barw sterowanych diod RGB. Funkcje *Załącz*, *Wyłącz* i *Przełącz* wyjście X umożliwiają zdalne sterowanie dodatkowym wyjściem opisanym na schemacie jako WYJ_X. Funkcje *Włącz sekwencję 1 i 2* załączają przykładowe pseudolosowe sekwencje świetlne RGB. Przyciski *„Załącz wszystko”* i *„Wyłącz wszystko”* umożliwiają załączenie wszystkich wyjść z maksymalnym wysterowaniem lub całkowite wygaszenie oświetlenia. Jeżeli nie chcemy, żeby któraś z dostępnych funkcji była aktywna, to jako parametr jej rozkazu wybieramy nastawę *„Nieaktywny”*. Po takim ustawieniu, funkcji tej nie da się wywołać. Jeżeli w którymś z pól, przeznaczonych do wprowadzania danych, tło zmieni kolor z odcienia zielonego na czerwony, oznacza to, że wpisana tam wartość jest niezgodna z dostępnymi opcjami. Po ewentualnej zmianie ustawień klikamy na przycisk *„Generuj program HEX dla ATTiny13”* i wszystkie nastawy zostaną wpisane w wygenerowany plik HEX, którym następnie programujemy mikrokontroler ATTiny13. Fusebity procesora powinny być ustawione zgodnie z komunikatem pojawiającym się po wygenerowaniu programu. Najważniejsze parametry to ustawienie pracy generatora RC na częstotliwość taktowania 9,6MHz i wyłączenie dzielenia zegara systemowego przez 8. Po prawidłowym zaprogramowaniu mikrokontrolera danymi z pliku HEX i ustawieniu fusebitów, układ powinien od razu prawidłowo działać. Wszystkie pliki pomocnicze oraz program dostępne są w Elportalu wśród materiałów dodatkowych do tego numeru EdW.

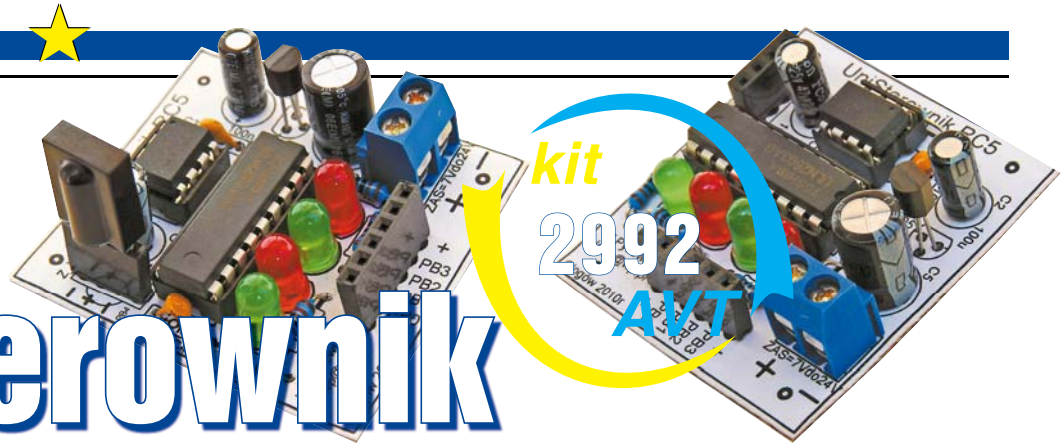
Wiesław Pytlewski
elewp@wp.pl

Komplet podzespołów z płytką jest dostępny w sieci handlowej AVT jako kit szkolny:
AVT2992 - UniSterownik,
AVT2992/2 - Płytka diod LED,
AVT2992/3 - Płytka przekaźnika.

Rys. 7



Rys. 8



UniSterownik

część 3 – Czterokanałowe zdalne sterowanie

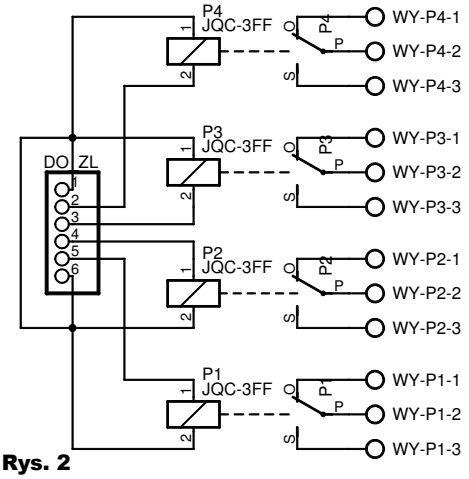
Prezentowana seria projektów **UniSterownik**, rozpoczęta w EdW 11/2011, przeznaczona jest przede wszystkim dla tych, którzy do tej pory nie odważyli się na kontakt z mikroprocesorami. Nie trzeba tu pisać kodu – programu dla mikroprocesora. Można ściągnąć z Elportalu gotową, przykładową wersję kodu. Ale znacznie lepiej będzie samodzielnie wygenerować „własny" kod na komputerze PC – Autor udostępnił stosowny program. Zadaniem początkującego użytkownika jest tylko wpisanie kodu do mikroprocesora i... cieszenie się działaniem tak uzyskanego układu. W części pierwszej opisano jak zbudować i zaprogramować UniSterownik. W części drugiej opisaliśmy aplikację do zdalnego sterowania diody RGB..

Jestem przekonany, że nie trzeba nikomu tłumaczyć, jakie zalety mają układy zdalnego sterowania. Wystarczy wspomnieć takie wynalazki, jak pilot do telewizora czy zdalnie sterowany centralny zamek. Dzisiaj już trudno sobie wyobrazić życie bez takich udogodnień. Myślę, że właśnie dlatego konstrukcje układów zdalnego sterowania cieszą się nieustannym zainteresowaniem elektroników. Zdalne sterowanie jest też jednym z najbardziej oczywistych zastosowań platformy UniSterownika. Układ może bezpośrednio sterować cztery obwody zasilane stałym napięciem o wartości do 30V i pobierające prąd do 1A. Można więc bezpośrednio z wyjść układu sterować małymi żaróweczkami, silniczkami, brzęczykami, diodami LED itp. Po podłączeniu do UniSterownika przekaźników, pracujących w funkcji elementów wykonawczych, układ może załączać dowolne urządzenia lub oświetlenie. Łatwa konfiguracja przycisków pilota, które będą sterować poszczególnymi funkcjami układu, jest dodatkowym atutem prezentowanego rozwiązania.

Jak to działa?

Działanie układu omawiałem już przy okazji opisu konstrukcji UniSterownika w EdW nr 11/2011. Podstawowym zadaniem układu elektrycznego, przedstawionego na schemacie z **rysunku 1**, jest oczekiwanie na transmisję z pilota RC5. Po odebraniu transmisji, dekodowane są dane zawarte w kodzie RC5. Następnie, zgodnie z otrzymanym poleceniem, odpowiednio zostają wysterowane wyjścia. Układem steruje mikrokontroler ATTiny13.

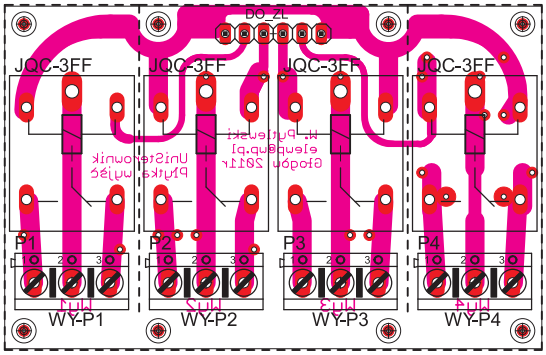
Wyjścia na złączu ZL sterowane są z wyprowadzeń PB0, PB1, PB2 i PB3 poprzez wzmacniacz ULN2803. Dzięki zamontowanym na płytce diodom LED możemy bezpośrednio obserwować jak są wysterowane wyjścia i czy dany przekaźnik jest załączony. Napięcie zasilania układu (U_ZAS) nie powinno przekraczać 30V. Jak wspomniałem wcześniej, wprost z płytki UniSterownika można sterować tylko obwodami o niewielkim obciążeniu. Dlatego opracowałem dodatkowy obwód z przekaźnikami umożliwiającymi sterowanie układów zewnętrznych o dużej mocy (lampy oświetleniowe, grzejniki, silniki, sygnalizatory i inne urządzenia, które wymagają do pracy dużego prądu lub zasilania z sieci). Na schemacie z **rysunku 2** rozrysowany jest sposób podłączenia przekaźników do układu UniSterownika. Diody antyprzepięciowe nie są tu konieczne, ponieważ są zawarte w kostce US3. Napięcie znamionowe cewek przekaźników powinno być takie jak napięcie zasilania U_ZAS. Najodpowiedniejsze będą przekaźniki z cewką na 12V lub 24V prądu stałego (DC).



Rys. 2

Montaż i uruchomienie

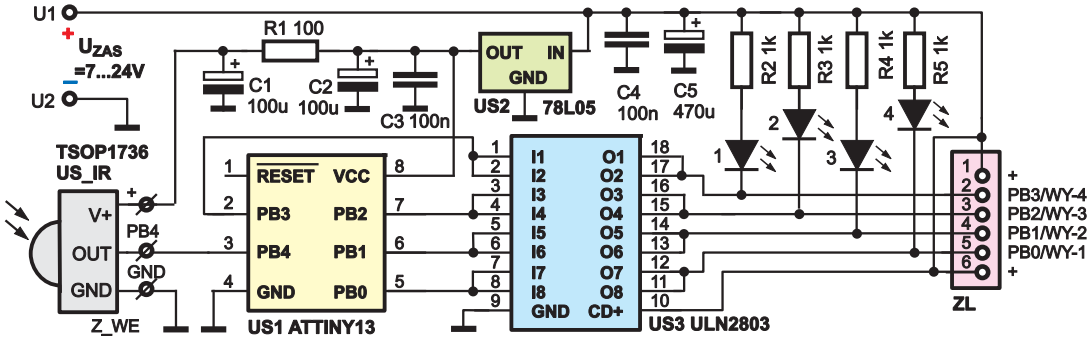
Sposób montażu płytki głównej sterownika opisałem przy okazji ogólnej prezentacji konstrukcji UniSterownika. Wspomnę tylko, że układ można zmontować na płytce drukowanej lub jak ktoś nie lubi lutować, to może uruchomić układ sterownika na płytce stykowej. Więcej szczegółów w EdW nr 11/2011.

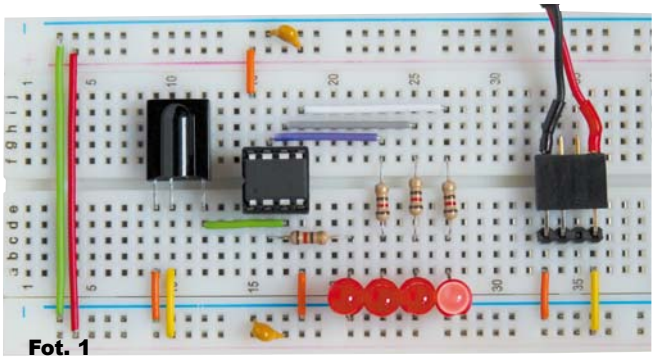


Rys. 3

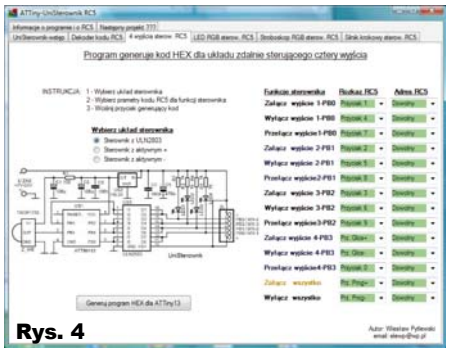
W celu rozbudowy UniSterownika o układ z przekaźnikami, zaprojektowana została specjalna płytka. Na **rysunku 3** widzimy rysunek płytki drukowanej, przystosowanej do monta-

Rys. 1





Fot. 1



Rys. 4

żu przekaźników JQC-3FF. Płytka przewidziana jest również do innych aplikacji, dlatego ma dodatkowe otwory pod inne elementy. Na rysunku 3 widoczny jest sposób zamontowania przekaźników i złącz wyjściowych. Jeżeli w danym układzie, nie jest wymagane zastosowanie czterech przekaźników, to po odcięciu zewnętrznych części można płytkę przerobić dla trzech lub dwóch przekaźników. Przy zastosowaniu płytki do sterowania obwodami zasilanymi z sieci trzeba uważać, ponieważ na ścieżkach od spodu płytki wystąpi niebezpieczne napięcie 230V.

Na **fotografii 1** przedstawiony jest widok najprostszej wersji zdalnego sterowania, zmontowanego na płytce stykowej. Płytka jest zmontowana według schematu z zakładki „*4 wyjścia sterow. RC5*" w programie „*ATTiny13 - UniSterownik RC5*". Schemat widoczny jest po wybraniu opcji „*Sterownik z aktywnym +*". Ta wersja układu, świetnie nadaje się do testów. Pilotem sterujemy tu załączanie diod świecących LED 1 do 4. Szczegółową fotografię układu zmontowanego na płytce stykowej można pobrać z Elportalu.

Generalnie zamiana UniSterownika w układ zdalnego sterowania polega na zmianie programu wpisanego do mikrokontrolera. Do zaprogramowania mikrokontrolera potrzebny jest plik HEX z danymi programu. Jednym z założeń do tego projektu było umożliwienie docelowemu użytkownikowi łatwego wyboru przycisków pilota, sterujących poszczególnymi funkcjami. Dlatego plik HEX generowany jest przez opisany już częściowo program konfigurujący „*ATTiny13-UniSterownik RC5*" – program dostępny jest na Elportalu. Po uruchomieniu programu na komputerze PC, wybieramy zakładkę „*4 wyjścia sterow. RC5*" – okno tej zakładki widoczne jest na **rysunku 4**.

Postępując zgodnie z instrukcją, w pierwszej kolejności wybieramy układ elektryczny sterownika. Do wyboru są trzy opcje. Układ pracy ze wzmacniaczem ULN2803, czyli identyczny z płytką UniSterownika. Pozostałe opcje to sterownik z aktywnym plusem i minusem. Opcje te przewidziane są dla konstrukcji, w których do mikrokontrolera trzeba będzie podłączyć inne wzmacniacze niż ULN2803 wymagające do wysterowania stanu niskiego (minusa) lub wysokiego (plusa).

Po prawej stronie okna znajduje się lista funkcji sterownika. Dostępne funkcje umożliwiają załączanie, wyłączanie lub przełączanie poszczególnych wyjść 1 do 4. Dzięki dwóm ostatnim funkcjom można załączać lub wyłączać wszystkie wyjścia na raz poprzez naciśnięcie jednego przycisku w pilocie. Mimo że rozkazy sterujące poszczególnymi funkcjami są wstępnie zadeklarowane, możemy im przypisać inny numer rozkazu i adresu, które będą odpowiadały wybranemu przyciskowi sterującemu z pilota. Jeżeli dana funkcja nie ma działać i nie chcemy, żeby sterownik ją wykonywał, to na liście rozkazów sterujących wybieramy opcję „*Nieaktywny*". Dane rozkazu i adresu zmieniamy, wybierając jedną z dostępnych opcji lub wpisując odpowiednią wartość. Jeżeli w którymś z pól przeznaczonych do wprowadzania danych tło zmieni kolor z odcienia zielonego na czerwony, oznacza to, że wpisana tam wartość jest niezgodna z dostępnymi opcjami. Po ustawieniu wszystkich opcji klikamy na przycisk „*Generuj program HEX dla ATTiny13*". Jeżeli program nie zgłosi żadnych błędów, to powinna się pojawić informacja o tym, jak się nazywa i gdzie został zapisany plik wynikowy HEX oraz jak należy zaprogramować Fusebity. Najważniejsze parametry to ustawienie pracy generatora RC na częstotliwość taktowania 9,6MHz i wyłączenie dzielenia zegara systemowego przez 8. Z Elportalu można pobrać również przykładowe pliki HEX wygenerowanych programów. Po zamontowaniu do UniSterownika zaprogramowanego mikroprocesora układ powinien od razu prawidłowo realizować polecenia wysłane z pilota RC5.

Wiesław Pytlewski
elewp@wp.pl

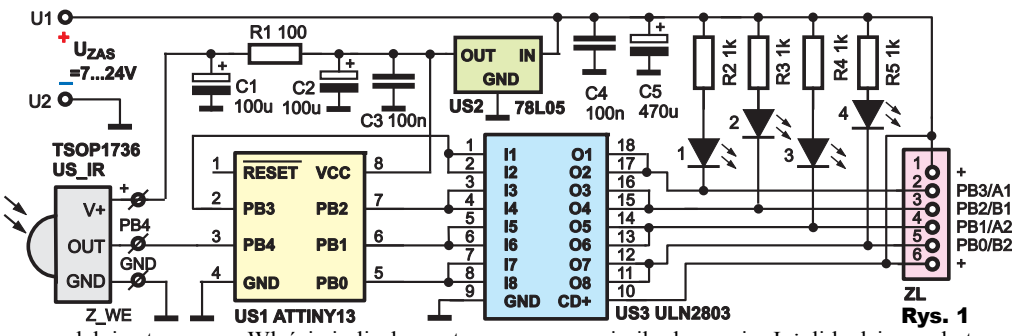
Komplet podzespołów z płytką jest dostępny w sieci handlowej AVT jako kit szkolny:
AVT2992 - UniSterownik,
AVT2992/2 - Płytka diod LED,
AVT2992/3 - Płytka przekaźnika,
AVT2992/4 - Czterokanałowe zdalne sterowanie.

UniSterownik
część 4
Sterownik silnika krokowego

Prezentowana seria projektów **UniSterownik**, rozpoczęta w EdW 11/2011, przeznaczona jest przede wszystkim dla tych, którzy do tej pory nie odważyli się na kontakt z mikroprocesorami. Nie trzeba tu pisać kodu – programu dla mikroprocesora. Można ściągnąć z Elportalu gotową, przykładową wersję kodu. Ale znacznie lepiej będzie samodzielnie wygenerować „własny" kod na komputerze PC – Autor udostępnił stosowny program. Zadaniem początkującego użytkownika jest tylko wpisanie kodu do mikroprocesora i... cieszenie się działaniem tak uzyskanego układu. W części pierwszej cyklu opisano jak zbudować i zaprogramować UniSterownik. W kolejnych artykułach opisane zostały aplikacje do sterowania kolorowej diody RGB i czterokanałowego zdalnego sterowania

Do czego to służy?

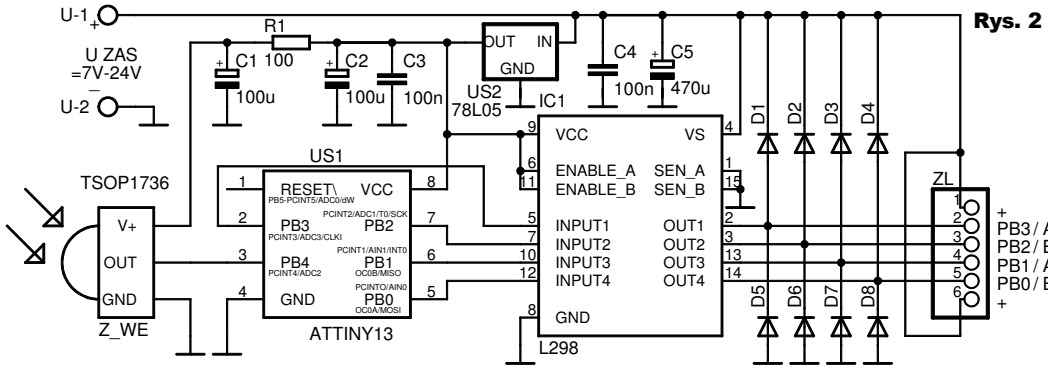
Jako kolejne z opracowanych zastosowań UniSterownika, chciałbym zaprezentować prosty układ zdalnego sterowania unipolarnym silnikiem krokowym. Warto zainteresować się silnikami krokowymi, ponieważ są one podstawowymi elementami napędowymi w amatorskich konstrukcjach robotów i maszyn CNC. Silniki krokowe w najprostszych aplikacjach nie wymagają skomplikowanych przekładni mechanicznych i łatwo da się je zastosować w precyzyjnych układach napędowych. Silniki te można pozyskać ze starego sprzętu komputerowego. Drukarki, skanery i stacje dysków zawierają nie tylko silniki krokowe. Urządzenia te są również źródłem wielu innych ciekawych i przydatnych elementów elektronicznych. Układ UniSterownika z dołączonym silnikiem krokowym może mieć wiele zastosowań praktycznych. Można na przykład przymocować bezpośrednio do osi silnika wieżyczkę modelu czołgu lub jakiś inny element, którym następnie będziemy mogli poruszać za pomocą zdalnego sterowania. Silnik można również zastosować do obracania kuli z naklejonymi lusterkami, wzbogacając paletę efektów świetlnych na imprezie tanecznej. Dodatkową zaletą takiego rozwiązania jest to, że możemy prędkość i kierunek obrotów kuli regulować zdalnie za pomocą pilota pracującego w standardzie RC5. W prosty sposób można sprząc silnik z potencjometrem wzmocnienia i w ten sposób uzyskamy zdalne sterowanie głośności we wzmacniaczu. Silnik krokowy ma tę zaletę, że nie stawia dużego oporu przy obracaniu jego osi, co umożliwia nastawianie potencjometru zdalnie lub ręcznie. Stosując dwa moduły UniSterownika i dwa silniki krokowe, można zbudować prosty pojazd zdalnie sterowany. Właściwie liczba zastosowań tej aplikacji UniSterownika, zależy tylko od naszej pomysłowości.



Rys. 1

Jak to działa?

Działanie układu omówiłem już wcześniej przy okazji opisu konstrukcji UniSterownika w EdW nr 11/2011. Podstawowym zadaniem układu elektrycznego, przedstawionego na schemacie z **rysunku 1** jest oczekiwanie na polecenia przekazywane z pilota transmisją RC5 i sterowanie silnikiem krokowym zgodnie z otrzymanymi poleceniami - więcej w opisie w EdW11/2011 i następnych numerach EdW. Sterowanie silnika krokowego polega na odpowiednim zasilaniu jego uzwojeń w taki sposób, żeby wytwarzały one wirujące pole magnetyczne. Wirnik silnika krokowego jest złożony z magnesów, które są przyciągane przez to wirujące pole magnetyczne, w wyniku czego wirnik obraca się. Jeżeli będziemy zbyt szybko przełączać napięcia na uzwojeniach silnika, to wirnik w końcu nie będzie nadążać za polem magnetycznym i się zatrzyma. Dlatego maksymalne prędkości, ogólnodostępnych silników krokowych są mniejsze niż 1000obr/min . Natomiast minimalna prędkość obrotowa silnika krokowego jest nieograniczona i może wynosić na przykład 1 obrót na rok. Zależy to tylko od sposobu sterowania silnikiem. Drugą istotną cechą silnika krokowego jest możliwość obracania nim o zadany, znany kąt. Ta właściwość daje nam możliwość precyzyjnego obracania takim silnikiem, co jest istotne w wielu zastosowaniach. Przy podłączaniu silnika do UniSterownika należy pamiętać, że maksymalny prąd wyjścia może wynosić do 1A .Diody LED dają możliwość obserwacji, w jaki sposób procesor steruje silnikiem krokowym. Gdy dana dioda świeci, odpowia-



Rys. 2

dające jej uzwojenie silnika jest załączone. Obserwując sekwencję zaświecania się i wyłączania kolejnych diod LED, łatwiej zrozumieć, w jaki sposób są przełączane – komutowane uzwojenia silnika krokowego. **Wprost z płytki UniSterownika można sterować tylko małymi silnikami unipolarnymi – mają one wyprowadzonych 5, 6 lub 8 przewodów.**

Ze względu na to, że w nowszych urządzeniach komputerowych coraz częściej stosowane są silniczki krokowe bipolarne, które mają 4 przewody do podłączenia, opracowałem drugą wersję układu. Różni się ona stopniem mocy wzmacniającym sygnały sterujące silnikiem. Jest to wzmacniacz L298, który może sterować silnikami bipolarnymi i unipolarnymi. Schemat elektryczny sterownika z układem L298 przedstawiony jest na **rysunku 2**.

Układ L298 jest specjalizowanym wzmacniaczem przeznaczonym do sterowania silników i to nie tylko silników krokowych, ale także zwykłych silniczków na prąd stały. Ze względu na małą liczbę portów układu ATTiny13, układ L298 pracuje tu tylko jako wzmacniacz sygnałów sterujących z mikrokontrolera. Wydajność prądowa poszczególnych wyjść wynosi 2A. Jego wyjścia, zależnie od wysterowania wejść, mogą być źródłem prądu lub jego odbiornikiem, czyli są wewnętrznie podłączane do plusa lub do minusa zasilania. Właściwość ta umożliwia sterowanie silników bipolarnych. Układ wymaga zabezpieczenia wyprowadzeń sterujących silnikiem za pomocą diod (na schemacie są to diody D1 do D8). Według noty informacyjnej układu L298 powinny to być szybkie diody o dość dużej wydajności prądowej.

Montaż i uruchomienie

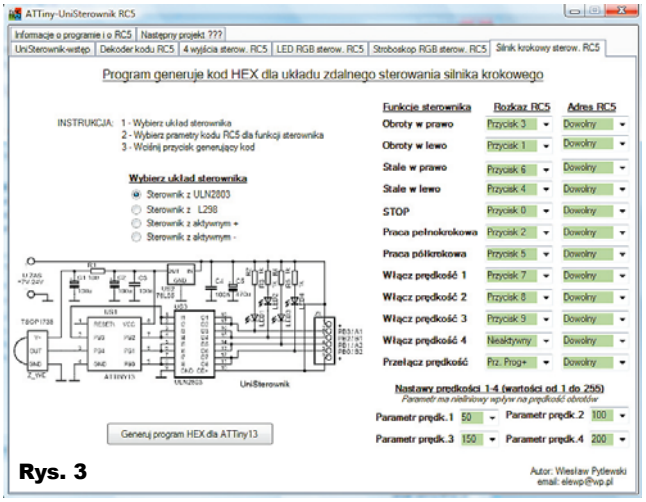
Sposób montażu układu opisałem przy okazji ogólnej prezentacji konstrukcji UniSterownika w EdW nr 11/2011. Nadmienię tylko, że układ można zmontować na płytce drukowanej lub też bez potrzeby lutowania sterownik z wzmacniaczem ULN można zmontować na płytce stykowej - fotografia dostępna w Elportalu

Oczywiście do poprawnej pracy mikrokontrolera niezbędny jest plik HEX, który wygenerowany zostanie przez program konfigurujący *ATTiny13-UniSterownik RC5-V2*. Program ten daje możliwość ustawienia parametrów prędkości silnika oraz możemy w nim ustalić, jakie przyciski w pilocie będą uruchamiały poszczególne funkcje UniSterownika. Po uruchomieniu programu na PC wybieramy zakładkę opisaną jako *Silnik krokowy sterow.RC5*. Widok okna tej zakładki przedstawiony jest na **rysunku 3.** W pierwszej kolejności wybieramy układ elektryczny sterownika. Do wyboru są cztery opcje. Pierwsza to układ pracy identyczny z płytką UniSterownika. Druga opcja to układ ze wzmacniaczem L298. Dwie ostatnie opcje to sterowniki z aktywnym plusem i aktywnym minusem. Te dwie ostatnie opcje umożliwiają wykorzystanie programu do sterowania układu wykonawczego opartego na dowolnym, innym układzie wzmacniacza lub na tranzystorach mocy.

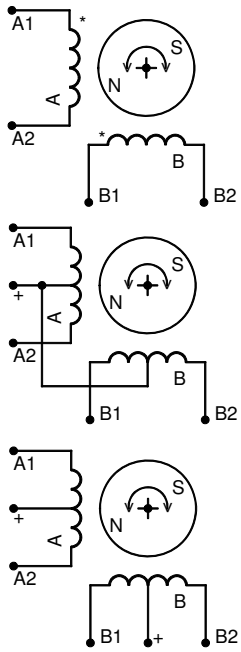
Rozkazy i adresy kodu RC5, sterujące poszczególnymi funkcjami sterownika, można skonfigurować w polach po prawej stronie okna. Pośród dostępnych możliwości pracy sterownika są funkcje załączania obrotów silnika w prawo i w lewo tylko na czas trzymania przycisku w pilocie oraz funkcje załączania obrotów w obydwu kierunkach na stałe. Poza tym można skonfigurować przycisk pilota sterujący funkcją STOP oraz przyciski przeznaczone do sterowania przełączaniem pracy silnika w tryb pełnokrokowy lub półkrokowy. Dalsze cztery funkcje powodują zmianę prędkości obrotu wału silniczka. Można wybierać jedną z czterech nastaw prędkości. Parametry dla poszczególnych prędkości od 1 do 4 wpisujemy w czterech polach do edycji tych wartości. Pola te znajdują się na dole pod listą funkcji sterownika. Jak zaznaczyłem w opisie, parametry nastaw prędkości mają nieliniowy wpływ na obroty silnika i powinny zawierać się w zakresie od 1 do 255. Ostateczne wartości dla parametrów prędkości trzeba ustalić doświadczalnie, zależnie od potrzeb i parametrów użytego silnika. Mikrokontroler po restarcie pobiera wartość pierwszego parametru prędkości do sterowania silnikiem. Jeżeli na przykład chcielibyśmy, aby sterownik obracał silnik tylko z jedną prędkością, to można wpisać wybraną wartość jako parametr wszystkich prędkości. Ostatnia funkcja do przełączania prędkości umożliwia przełączanie jednym przyciskiem obrotów silnika – cyklicznie od prędkości 1 do 4. Jeżeli nie chcemy, aby można było załączyć jakąś z dostępnych funkcji, to jako jej parametr wybieramy nastawę *Nieaktywny*. Gdy wybierzemy układ elektryczny sterownika i opcje przycisków sterujących, program wygeneruje plik HEX do zaprogramowania procesora. Fusebity procesora powinny być ustawione na opcję załączającą zegar wewnętrzny RC 9,6MHz, a dzielenie zegara systemowego przez 8 powinno być wyłączone.

Po zaprogramowaniu układu i podłączeniu zasilania, oraz co jest bardzo ważne, właściwym podłączeniu silnika, układ powinien od razu działać. Tu może pojawić się wątpliwość, jak podłączyć silnik do tego sterownika? Na **rysunkach 4a, 4b i 4c** widoczne są trzy układy silników krokowych, mają one 4, 5 lub 6 przewodów do podłączenia. Przy ustalaniu wyprowadzeń najlepiej posłużyć się omomierzem.

W silniku z 4 przewodami (rys. 4a) łatwo jest ustalić cewkę A i B. Są to dwie jednakowe cewki, niepołączone ze sobą. Wyprowadzenia



Rys. 3



Rys. 4a, b, c

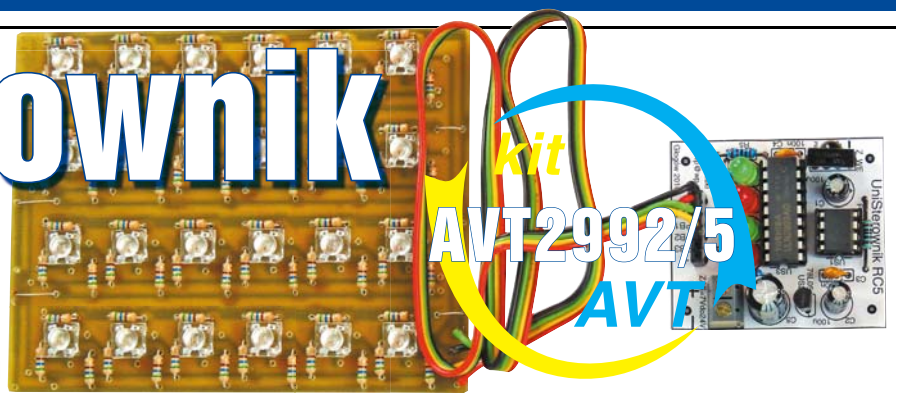
cewki A podłączamy do A1 i A2 w złączu ZL. Cewkę B podłączamy do B1 i B2. **Silnik z czterema przewodami może być sterowany tylko przez układ ze wzmacniaczem L298** lub podobnym. W silniku z 5 przewodami (rys4b) ustalamy przewód wspólny i podłączamy go do „+" w złączu ZL. Między przewodem wspólnym a resztą przewodów jest jednakowa rezystancja. Pozostałe cztery wyprowadzenia podłączamy do ZL, eksperymentując tak, aby silnik obracał się płynnie i bez „dziwnych" i nieregularnych skoków. W silniku z 6 przewodami (rys4c) ustalamy przewody wspólne obydwu zespołów cewek i podłączamy je do „+" w złączu ZL. Następnie wyprowadzenia cewek A i B podłączamy odpowiednio do A1, A2 i B1, B2 złącza ZL.

Jeżeli po załączeniu układu silnik będzie kręcił się w prawo zamiast w lewo, to trzeba podłączyć odwrotnie złącze silnika do gniazda ZL (obracamy je o 180 stopni).

UWAGA! W Elportalu, dostępna jest nowa, poprawiona wersja programu, generującego kody HEX do aplikacji UniSterownika – „ATTiny13-UniSterownik RC5-V2". Wyróżnia się ona dopiskiem "V2" w tytule i na ikonie programu.

Wiesław Pytlewski
elewp@wp.pl

UniSterownik
część 5
Stroboskop RGB

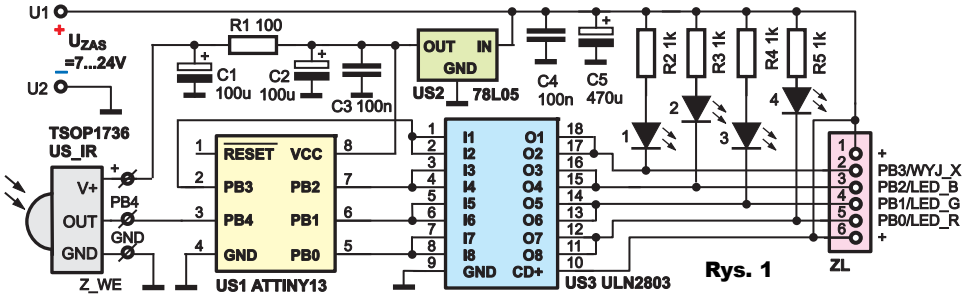


Tym razem program przekształci UniSterownik w układ dyskotekowego efektu świetlnego. Jednym z popularnych efektów świetlnych, używanych na imprezach, jest oświetlenie stroboskopowe. Krótkie błyski światła, o odpowiedniej częstotliwości, dają pozorne wrażenie spowolnienia ruchu oświetlanego przedmiotu, dlatego zaczęto je wykorzystywać jako efekty dyskotekowe. Nasz układ stroboskopu jest bezpieczny w użytkowaniu, ponieważ zastosowanie diod LED RGB wykluczyło potrzebę zasilania wysokim napięciem. Ich użycie umożliwia uzyskanie siedmiu barw emitowanego światła. Poza pracą w trybie stroboskopu, układ również generuje sekwencje błysków na przemian z różnokolorowym oświetleniem lub też może odgrywać rolę kolorowego oświetlacza. Stroboskop ten jest urządzeniem zdalnie sterowanym za pomocą pilota RC5, dzięki czemu można go umieścić z dala od konsoli didżeja, unikając rozkładania dodatkowych przewodów do sterowania.

Prezentowane dotychczas zastosowania UniSterownika mogą być bardzo pomocne w budowie różnych efektów świetlnych. UniSterownik jako sterownik silnika krokowego może kręcić z różnymi prędkościami lustrzaną kulą lub obracać różnokolorowym filtrem umieszczonym przed halogenowym reflektorem. UniSterownik w aplikacji czterokanałowego zdalnego sterowania może zdalnie załączać oświetlacze lustrzanej kuli lub inne kolorowe oświetlenie. Wszystkimi tymi urządzeniami możemy sterować za pomocą jednego pilota pracującego w standardzie RC5.

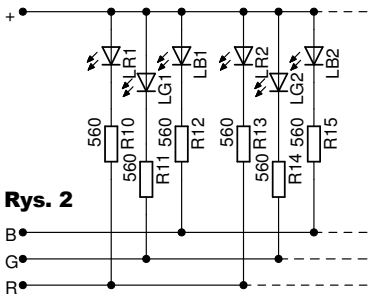
Opis układu

Działanie układu elektrycznego UniSterownika opisane było w EdW 11/2011. Pod względem elektrycznym układ stroboskopu jest identyczny z opisanym w EdW nr 12/2011 sterownikiem kolorowej diody RGB. Aplikacje te różnią się tylko programem wpisanym do mikrokontrolera. Schemat przedstawiony jest na **rysunku 1**. Przypomnę tylko, że ATTiny13 odbiera sygnały RC5 z odbiornika podczerwieni TSOP1736 i steruje wyjściami. Wyjścia mogą sterować obwodami, zasilanymi napięciem stałym do 30V i być obciążane prądem do 1A. W aplikacji stroboskopu do wyjść podłączamy zespolone diody RGB ze wspólną anodą (CA – Common Anode) i pojedyncze diody. Możemy podłączać do wyjść jednocześnie różne typy diod, ewentualnie taśmy, pamiętając, że prąd sterowany przez pojedyncze wyjście nie może przekroczyć 1A. Zastosowane diody powinny być przezroczyste typu „clear".



Rys. 1

Na **rysunkach 2, 3 i 4** widoczne są sposoby dołączania różnych rodzajów diod do wyjść UniSterownika. Do sterowania diod RGB z wspólną katodą trzeba użyć dodatkowych tranzystorów sterujących (PNP).



Rys. 2

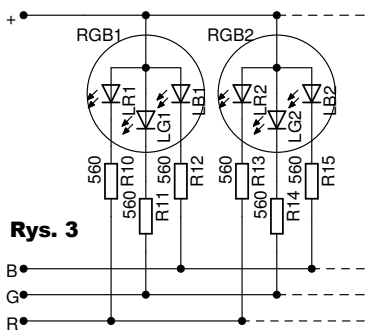
Wartości rezystorów dla pojedynczych diod lub zestawu diod wyliczamy z wzoru:

R=(U_ZAS – U_DIOD)/I_DIODY

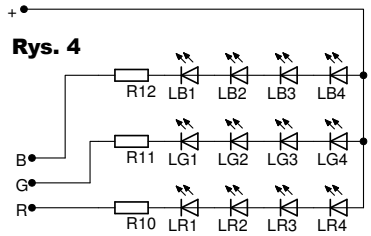
Na **rysunku 5** przedstawiony jest schemat dołączenia przekaźnika do zdalnie sterowanego wyjścia WYJ_X. Na **rysunku 6** widoczna jest płytka drukowana przeznaczona do montażu przekaźnika JQC-3FF.

Montaż i uruchomienie

Montaż UniSterownika opisany jest szczegółowo w EdW nr 11/2011. Na rynku dostępnych jest wiele różnego rodzaju taśm i listew ze świecącymi diodami LED, które mogą być podłączone wprost do złącza ZL UniSterownika. Wtedy napięcie zasilania układu (U_ZAS) powinno być takie, jak napięcie zasilania użytej taśmy LED (12 lub 24V). Można również zastosować zaprojektowaną do tego płytkę oświetlacza (**rysunek 7**), na której da się zamocować zarówno zespolone diody RGB typu SUPER FLUX, jak i pojedyncze diody o średnicy 5mm (**fotografia 1**). Kilka płytek z rysunku 7 może stworzyć większy zespół. Do budowy większego, zespolonego oświetlacza polecam płytkę zaprezentowaną na **rysunku 8**, na której mieszczą się 24 zespolone diody SUPER FLUX. Projekty płytek (EAGLE) dostępne są na Elportalu.



Rys. 3

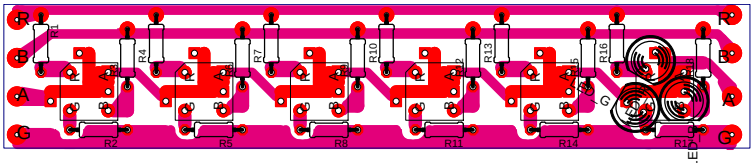


Rys. 4

Oświetlacz z 20–30 zespolonych diod RGB typu SUPER FLUX nadaje się do pracy w małych pomieszczeniach i jest odpowiedni na domową prywatkę. UniSterownik może wysterować znacznie mocniejsze zespoły diod (Imax=1A Umax=30V). Wykonałem kilka eksperymentów z diodami LED o mocy 1W, które pobierają prąd 350mA. Już przy zastosowaniu po jednej kolorowej diodzie na kanał efekt jest bardzo ciekawy. Po rozstawieniu diod w pewnych odległościach od siebie, poza błyskami światła, można zaobserwować różnokolorowe cienie rzucane przez oświetlane z różnych stron przedmioty. Stosując diody mocy 1- lub 3-watowe, musimy pamiętać, że rezystory współpracujące z tymi diodami muszą mieć również odpowiednią moc. Moc (P) zastosowanego rezystora określamy z zależności:

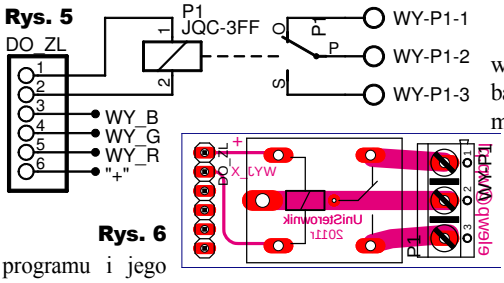
P=(U_ZAS – U_DIOD)*I_DIODY.

Na **fotografii 2** przedstawiony jest widok eksperymentalnego układu stroboskopu RGB z jedną diodą. Układ złożony na płytce stykowej służy do tego, by zorientować się, jakie jest ogólne działanie



Rys. 7

Fot. 1



Rys. 5

Rys. 6

programu i jego funkcji. Układ zmontowany jest zgodnie ze schematem z zakładki „Stroboskop RGB sterow. RC5" w programie „ATTiny13 – UniSterownik RC5-V2". Schemat widoczny jest po wybraniu opcji *Stroboskop LED RGB na wy aktywny - (WA)*. W podobnym układzie możemy również przetestować działanie układu z zespoloną diodą RGB mającą wspólną katodę (WK lub CC). W tym celu wybieramy w programie opcję *Stroboskop LED RGB na wy aktywny+(WK)* i programem wygenerowanym w tej opcji, programujemy procesor. Dioda po lewej stronie załączana jest poleceniami sterującymi wyprowadzeniem WYJ_X.

Do zaprogramowania mikrokontrolera potrzebny jest plik HEX z danymi programu. Plik taki generowany jest przez prezentowany już wcześniej program *ATTiny13 - UniSterownik RC5-V2*. Po uruchomieniu programu klikamy na zakładkę *Stroboskop RGB sterow. RC5*. Widok okna tej zakładki przedstawiony jest na **rysunku 9**.

Postępując zgodnie z instrukcją, w pierwszej kolejności wybieramy układ elektryczny sterownika. Do wyboru są trzy opcje. Układ pracy identyczny z płytką UniSterownika z wzmacniaczem ULN2803 oraz dwa sterowniki z aktywnym plusem lub aktywnym minusem na wyjściach. Po wybraniu układu pracy sterownika możemy skonfigurować przyciski pilota sterujące poszczególnymi funkcjami stroboskopu. Działanie poszczególnych funkcji jest następujące. Przyciski przypisane funkcjom *Zał/Wył błysk* powodują dodanie lub wygaszenie wybranego koloru ze składowej barw błysku. Dzięki tym funkcjom błyski mogą mieć jedną z siedmiu barw. Błyski mogą być emitowane z wygaszonego, ciemnego reflektora, ale można też *załączać / wyłączać tło* błysku, czyli pilotem będzie można ustawić pracę stroboskopu tak, że na przykład lampa świeci na niebiesko i w momencie błysku zmienia na moment kolor na biały. Przyciski przypisane do funkcji *Włącz sekwencję* 1, 2 lub 3 załączają jeden z trzech zaprogramowanych efektów świetlnych, złożonych z kombinacji sekwencji błysków. Są to proste przykłady możliwości i zalet stroboskopu zbudowanego z diod RGB i warto je przetestować. Przyciski *Częstość błysku* „+" i „–" zmieniają nastawę czasu przerwy pomiędzy błyskami na jedną z szesnastu predefiniowanych wartości. Przycisk *Zmień czas błysku* zmienia długość czasu błysku, który może mieć jedną z dwu wartości. Następną dostępną funkcją jest możliwość przełączenia pracy układu z trybu stroboskopu na tryb oświetlacza i z powrotem, co wywołujemy przyciskiem pilota zdefiniowanym przy funkcji *Przeł.strob/oświetlacz*. W trybie oświetlacza można oczywiście również załączać lub wyłączać poszczególne kolory składowe oświetlacza przyciskami funkcji Zał/Wył błysk. Nastawy kolorów oświetlacza i stroboskopu są zapamiętywane w oddzielnych zmiennych, co umożliwia szybki powrót do pracy w trybie stroboskopu i odwrotnie. Funkcja *Zał/Wył wyjście X* zmienia stan – przełącza dodatkowe wyjście opisane na schemacie, jako WYJ_X. Przycisk *Wstrzymaj/Zał. Pracę* wygasza lub z powrotem uruchamia stroboskop, przywracając parametry pracy. Przycisk *Restart-ustaw. pocz.* powoduje ustawienie początkowych parametrów i nastaw stroboskopu, dając możliwość zdalnego restartu programu. Funkcja ta jest przydatna, gdy pogubimy się, jakie tryby pracy mamy załączone. Wtedy czasem łatwiej zacząć od restartu. Jeżeli nie chcemy, żeby któraś z dostępnych funkcji była aktywna, to jako jej parametr wybieramy nastawę *Nieaktywny*. Po kliknięciu na przycisk *Generuj program…* zostanie utworzony plik HEX, zawierający wprowadzone nastawy. Plikiem tym programujemy procesor ATTiny13. Fusebity procesora powinny ustawiać taktowanie na zegar wewnętrzny RC 9,6MHz i wyłączać dzielenie zegara systemowego przez 8.

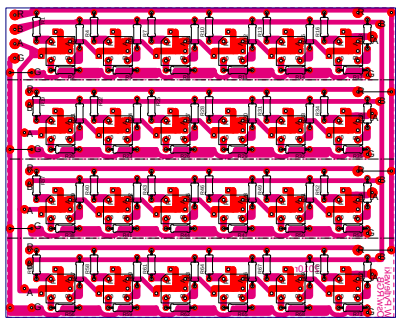
Zaprogramowany mikrokontroler montujemy na płytce UniSterownika i po dołączeniu diod układ jest gotowy do pracy. Po załączeniu zasilania czerwone diody kilka razy mignął i sterownik będzie oczekiwał na polecenia przesłane z pilota RC5, którym można zdalnie uruchamiać poszczególne funkcje stroboskopu.

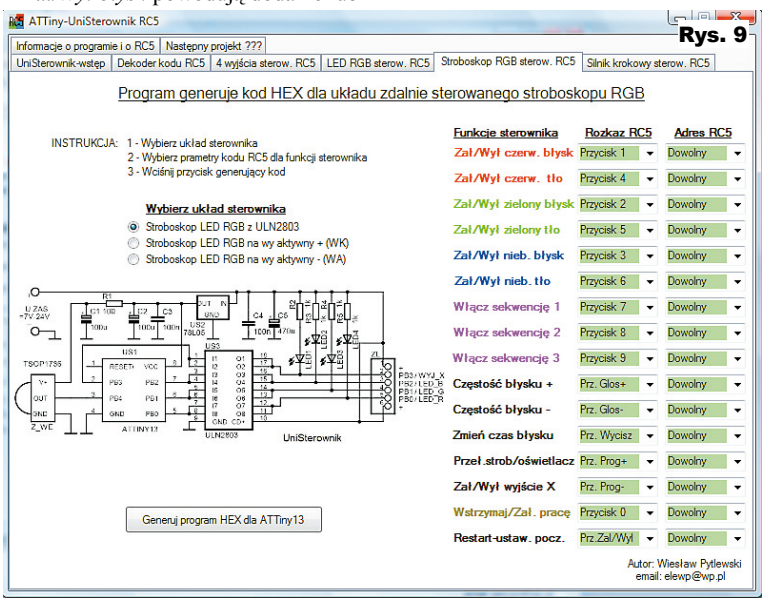
Ponieważ jest to ostatni artykuł z przygotowanego cyklu, chciałbym zwrócić uwagę na to, że zaprezentowane zastosowania UniSterownika to tylko kilka z możliwych aplikacji. Układ ten może również pracować jako regulator temperatury, sterownik oświetlenia, sterownik zamka, włącznik czasowy i może mieć jeszcze wiele, wiele innych zastosowań. Taką różnorodną funkcjonalność układ uzyskał dzięki zastosowaniu mikrokontrolera. Praca układu zależy właściwie tylko od programu zawartego w mikrokontrolerze. Może w przyszłości uda mi się zaprezentować jeszcze inne praktyczne zastosowania UniSterownika.

Wiesław Pytlewski
elewp@wp.pl

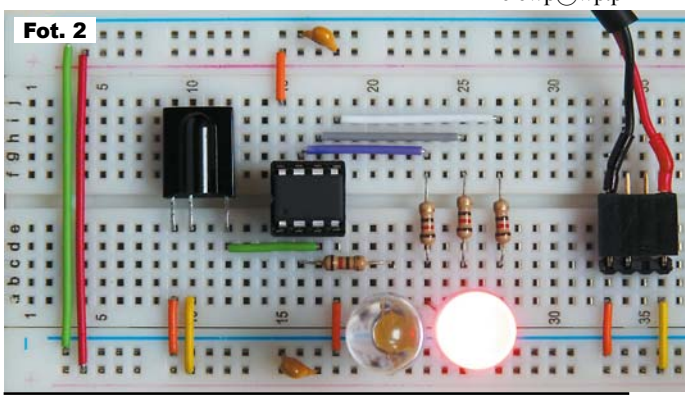
Komplet podzespołów z płytką jest dostępny w sieci handlowej AVT jako kit szkolny:
AVT2992 - UniSterownik,
AVT2992/2 - Płytka diod LED,
AVT2992/3 - Płytka przekaźnika,
AVT2992/4 - Czterokanałowe zdalne sterowanie,
AVT2992/5 - Stroboskop RGB.

Rys. 8. Skala 50%





Rys. 9



Fot. 2